

**REMARKS**

**Introduction**

Claims 1-15 are pending in this application, of which claims 1, 5 and 7-11 being independent. Claims 5, 6, 8, 10 and 15 have been withdrawn from consideration for prosecution of the present application. Claims 1, 3, 7, 9 and 11-14 have been amended to clarify aspects of the present application. Amendments to the claims are supported on page 11, line 18 – page 12, line 3; page 15, line 24 – page 16, line 24; and Figure 5 of the present application. Care has been exercised not to introduce new matter.

**Information Disclosure Statement**

In the instant Office Action, the Examiner asserts that the information disclosure statement filed September 4, 2008 fails to comply with 37 C.F.R. 1.98(a)(2) because it does not include a legible copy of each cited foreign patent document. Applicants respectfully request that JP 2000-148783, JP 2000-298637, JP 2001-160024 and JP 2001-209568 be considered because Applicants note that copies with English abstracts of JP 2000-148783, JP 2000-298637, JP 2001-160024 and JP 2001-209568 were submitted on September 4, 2008. Accordingly, Applicants respectfully request that the Examiner consider the above identified information disclosure statement and return the initialed copy of form PTO-1449 in the next official action.

Further, Applicants respectfully request that the Examiner consider “X/Open CAE Specification” listed on the information disclosure statement filed February 17, 2004 and return the initialed copy of the form PTO-1449 in the next official action.

For the Examiner’s convenience, Applicants are submitting the best available copies of the above documents together with this document.

**Claim Rejections – 35 U.S.C. § 103**

Claims 1-2, 7, 9 and 11-12 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard (U.S. 5,729,689) and further in view of Sitaraman (US 6,427,170). Claims 3 and 13 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard and Sitaraman as applied to claims 1 and 11, and further in view of BenShaul (US Pub. 2002/0010798). Claims 4 and 14 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Allard, Sitaraman and BenShaul as applied to claims 3 and 13, and further in view of Beser (US 6,754,622).

Applicants respectfully traverse these rejections for at least the following reasons.

Applicants respectfully submit that none of the cited references discloses or suggests the features as recited by amended claims 1, 7, 9 and 11, specifically “each of a plurality of nodes having system definition for other nodes.” In the claimed subject matter, a name information retrieval request is arranged to be transmitted to each of the plurality of nodes respectively, each of the nodes having its system definition (see, for example, at page 15, line 24 to page 16, line 24 of the specification). Then, the name information for each of the nodes is acquired.

Applicants respectfully submit that none of the cited references discloses the above discussed limitations. The Examiner asserts that Allard discloses a [third] node designated by a system definition at col. 16, lines 4-15 of Allard. However, Allard fails to disclose that the node is designated **by the system definition**. Further, Sitaraman appears to disclose a transmission operation of retrieval request toward each of the plurality cache memories. However, Sitaraman fails to disclose that each of the plurality of nodes has a system definition. In addition, other cited reference also fails to disclose the above discussed limitations of claims 1, 7, 9 and 11.

As such, it is clear that none of the cited references, taken alone or in any combination thereof, renders claims 1, 7, 9 and 11 or any claims dependent thereupon obvious. Applicants

**Application No.: 10/696,563**

respectfully request that the Examiner withdraw the rejection of claims 1-4, 7, 9 and 11-14 under 35 U.S.C. § 103(a).

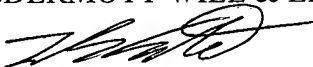
**CONCLUSION**

Having fully responded to all matters raised in the Office Action, Applicant submits that all claims are in condition for allowance, an indication for which is respectfully solicited. If there are any outstanding issues that might be resolved by an interview or an Examiner's amendment, the Examiner is requested to call Applicant's attorney at the telephone number shown below.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP



Takashi Saito  
Limited Recognition No. L0123

600 13<sup>th</sup> Street, N.W.  
Washington, DC 20005-3096  
Phone: 202.756.8000 TS:MaM  
Facsimile: 202.756.8087  
**Date: February 25, 2009**

**Please recognize our Customer No. 20277  
as our correspondence address.**

Docket No.: 062807-0146

COPY

PATENT

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

In re Application of	:	Customer Number: 20277
	:	
Makoto KOIKE, et al.	:	Confirmation Number: 5912
	:	
Application No.: 10/696,563	:	Group Art Unit: 2152
	:	
Filed: October 30, 2003	:	Examiner: J. Chang
	:	
For: SYSTEM AND METHOD FOR PROVIDING NAMING SERVICE IN DISTRIBUTED PROCESSING SYSTEM		

**INFORMATION DISCLOSURE STATEMENT**

Mail Stop Amendment  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Dear Sir:

In accordance with the provisions of 37 C.F.R. 1.56, 1.97 and 1.98, the attention of the Patent and Trademark Office is hereby directed to the references listed on the attached form PTO-1449. It is respectfully requested that the references be expressly considered during the prosecution of this application, and that the references be made of record therein and appear among the "References Cited" on any patent to issue therefrom.

This Information Disclosure Statement is being filed within three months of the U.S. filing date OR before the mailing date of a first Office Action on the merits. No certification or fee is required.



The following are brief comments on the subject matter of the five listed items.

- 1) Item 1 discloses a technique regarding a so-called wide area load dispersion system and method for performing load dispersion and failure avoidance between each of the pair of servers respectively provided within each of the networks different from each other.
- 2) Item 2 discloses a name resolver for resolving a name of host for a space including a plurality of domain names.
- 3) Item 3 discloses a management selecting method for a server application by which a client application is arranged to appropriately select an available server, and to connect it to a network.
- 4) Item 4 discloses an apparatus and a method for a directory service by which processing load can be made reduced when the whole system is synchronized.
- 5) With respect to Item 5, descriptions in paragraph [0019] can be read as “Fig. 3 shows one example of an arrangement of a transmitter-side directory server 101. In a directory memory unit 302, directories with structures shown in Fig. 2 can be stored. A directory operating unit 301 is arranged to add data into the directory memory unit 302, and to delete data having been stored in the directory memory unit 302. Information on an updating operation performed by the directory operating unit 301 can be supplied to an updated information generation unit 303. At the updated information generation unit 303, updated information can be generated in response to an updating operation, which updated information is used for copying the updated data of a directory to another directory having been provided within a pair of receiver-side directory servers 102 and 102”.

Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

10/696,563

Respectfully submitted,

McDERMOTT WILL & EMERY LLP

*to Brian K. Just Reg. No. 51,321*  
Keith E. George  
Registration No. 34,111

600 13<sup>th</sup> Street, N.W.  
Washington, DC 20005-3096  
Phone: 202.756.8000 KEG:mjb  
Facsimile: 202.756.8087  
**Date: September 3, 2008**

**Please recognize our Customer No. 20277  
as our correspondence address.**

COPY

SHEET 1 OF 1

<b>INFORMATION DISCLOSURE STATEMENT BY APPLICANT</b>  (Substitute for form 1449/PTO)				ATTY. DOCKET NO. <b>062807-0146</b>		SERIAL NO. <b>10/696,563</b>	
				APPLICANT <b>Makoto KOIKE, et al.</b>			
				FILING DATE <b>October 30, 2003</b>		GROUP <b>2152</b>	
<b>U.S. PATENT DOCUMENTS</b>							
EXAMINER'S INITIALS	CITE NO.	Document Number Number-Kind Code <sup>2</sup> (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document		Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	
		US					
		US					
		US					
<b>FOREIGN PATENT DOCUMENTS</b>							
EXAMINER'S INITIALS	CITE NO.	Foreign Patent Document Country Codes-Number 4-Kind Codes (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document		Pages, Columns, Lines Where Relevant Figures Appear	Translation Yes      No
	1	JP-A-2001-092766	04/06/2001	HITACHI LTD			JAPAN (w/ English abstract)
	2	JP-A-2000-148783	05/30/2000	NEC CORP			JAPAN (w/ English abstract)
	3	JP-A-2000-298637	10/24/2000	KYUSYU-NEC			JAPAN (w/ English abstract)
	1	JP-A-2001-160024	06/12/2001	NEC CORP			JAPAN (w/ English abstract)
	5	JP-A-2001-209568	08/03/2001	SONY CORP			JAPAN (w/ English abstract)
<b>OTHER ART (Including Author, Title, Date, Pertinent Pages, Etc.)</b>							
EXAMINER'S INITIALS	CITE NO.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.					
EXAMINER				DATE CONSIDERED			

\*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

1 Applicant's unique citation designation number (optional). 2 Applicant is to place a check mark here if English language Translation is attached.

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-298637

(43)Date of publication of application : 24.10.2000

(51)Int.Cl.

G06F 13/00

G06F 15/177

(21)Application number : 11-108156

(71)Applicant : NEC SOFTWARE KYUSHU LTD

(22)Date of filing : 15.04.1999

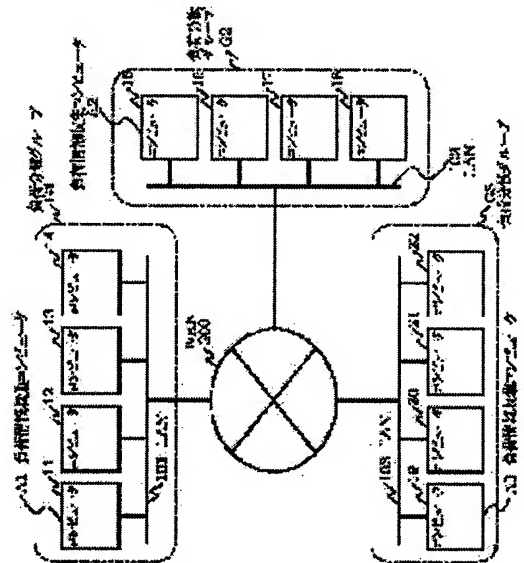
(72)Inventor : TANAKA KIYOHARU

## (54) SYSTEM AND METHOD FOR LOAD DISTRIBUTION AND RECORDING MEDIUM

### (57)Abstract:

**PROBLEM TO BE SOLVED:** To perform load distribution by grouping computers connected by a LAN(local area network), a WAN(wide area network), etc., and allowing the representative computer of the group to manage the load information of its own group and other groups.

**SOLUTION:** A load information collection computer A1 (computer 11) collects the loads of computers 11 to 14 in a load distribution group G1, sorts them in order of a low load, and further, prepares and stores average load information of its own group. It also acquires the average load information of respective groups from load information collection computers A2 and A3. When the computer A1 receives a processing request from a computer in its own group, it transfers the processing request to a group whose average load is lowest. In the case the group is its own group, it transfers the processing request to a computer whose load is lowest in the group.



(3)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-298637

(P2000-298637A)

(43) 公開日 平成12年10月24日 (2000. 10. 24)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	テーマコード* (参考)
G 0 6 F 13/00	3 5 7	G 0 6 F 13/00	3 5 7 Z 5 B 0 4 5
15/177	6 7 2	15/177	6 7 2 A 5 B 0 8 9
	6 7 4		6 7 4 B

審査請求 有 請求項の数 7 O L (全 11 頁)

(21) 出願番号 特願平11-108156

(22) 出願日 平成11年4月15日 (1999. 4. 15)

(71) 出願人 000164449

九州日本電気ソフトウェア株式会社

福岡市早良区百道浜2丁目4-1 NEC

九州システムセンター

(72) 発明者 田中 清春

福岡県福岡市早良区百道浜2-4-1 九

州日本電気ソフトウェア株式会社内

(74) 代理人 100082935

弁理士 京本 直樹 (外2名)

Fターム(参考) 5B045 GG04 GG09 JJ08

5B089 GA01 HA06 JA36 JB18 KA06

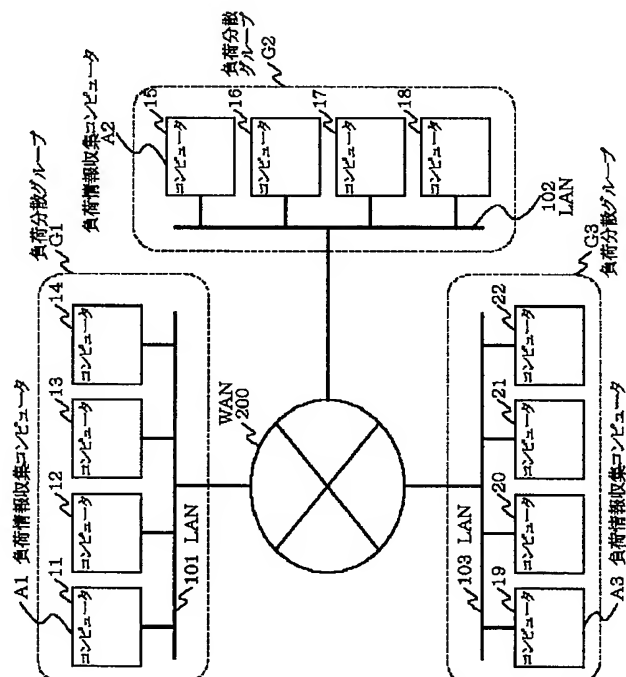
KB00 KB04 MA03

(54) 【発明の名称】 負荷分散システム、負荷分散方法、および記録媒体

(57) 【要約】

【課題】 LAN、WAN等で接続されたコンピュータをグループ化し、グループの代表のコンピュータが自グループ、他グループの負荷情報を管理し、負荷の分散をはかる。

【解決手段】 負荷情報収集コンピュータA1 (コンピュータ11) は、負荷分散グループG1内のコンピュータ11~コンピュータ14の負荷を収集し、負荷の低い順にソートし、さらに、自グループの平均負荷情報を作成し保持する。また、負荷情報収集コンピュータA2、負荷情報収集コンピュータA3から、それぞれのグループの平均負荷情報を取得する。負荷情報収集コンピュータA1は、自グループ内のコンピュータから処理要求を受け付けると、平均負荷の最も低いグループに対して、その処理要求を転送する。自グループであれば、自グループ内の最も負荷の低いコンピュータに処理要求を転送する。



**【特許請求の範囲】**

【請求項 1】 1 以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムであって、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの 1 つを負荷情報収集コンピュータとし、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第 1 の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第 2 の平均負荷情報を収集し、前記第 1 の平均負荷情報、および第 2 の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持する前記負荷情報収集コンピュータを有することを特徴とする負荷分散システム。

【請求項 2】 自身が属する前記負荷分散グループ内の前記コンピュータの負荷がある一定値を超えた場合に、対応する前記コンピュータの情報を前記負荷情報から削除する前記負荷情報収集コンピュータを有することを特徴とする請求項 1 記載の負荷分散システム。

【請求項 3】 前記ローカル・エリア・ネットワークに接続され、1 つの前記コンピュータと同等のレベルとして扱われる下位の階層の前記ローカル・エリア・ネットワークを有することを特徴とする請求項 1 または 2 記載の負荷分散システム。

【請求項 4】 自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送する前記負荷情報収集コンピュータを有することを特徴とする請求項 1、2 または 3 記載の負荷分散システム。

【請求項 5】 1 以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの 1 つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第 1 の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第 2 の平均負荷情報を収集し、前記第 1 の平均負荷情報、および第 2 の平均負荷情報に基づいて、負

荷の低い順位でソートしグループ負荷情報として保持させる手順を含むことを特徴とする負荷分散方法。

【請求項 6】 前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むことを特徴とする請求項 5 記載の負荷分散方法。

【請求項 7】 1 以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの 1 つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第 1 の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第 2 の平均負荷情報を収集し、前記第 1 の平均負荷情報、および第 2 の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させ、かつ、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むプログラムを記録することを特徴とする記録媒体。

**【発明の詳細な説明】****【0001】**

【発明の属する技術分野】 本発明は、負荷分散システム、負荷分散方法、記録媒体に関し、特に、ネットワークで接続された複数のコンピュータに対する負荷を分散させる負荷分散システム、負荷分散方法、記録媒体に関する。

**【0002】**

【従来の技術】 グループ化されたコンピュータまたはプロセッサに対する負荷分散の従来技術としては、たとえば、「特開平 7-93265 号公報」記載の技術がある。この技術は、共有メモリに接続された最下層の複数のプロセッサの負荷をこの共有メモリに書き込み、さらに、この共有メモリに接続された通信制御プロセッサがこの負荷を参照して負荷分散を実施するものである。さらに、負荷分散においては、乱数が使用される。

【0003】

【発明が解決しようとする課題】 上述した従来の技術の第1の問題点は、通信制御プロセッサが必要となることである。

【0004】 その理由は、最下層のプロセッサ同士がローカル・エリア・ネットワークで接続される構成となっていないからである。

【0005】 第2の問題点は、共有メモリが必要となることである。

【0006】 その理由は、共有メモリを介して通信制御プロセッサが負荷を参照する構成となっているからである。

【0007】 第3の問題点は、負荷分散の精度が悪いことである。

【0008】 その理由は、乱数を使用して負荷分散を行うからである。

【0009】 第4の問題点は、負荷分散のグループが任意に設定できないことである。

【0010】 その理由は、同一の階層レベル同士でしかグループを構成できないからである。

【0011】 本発明の目的は、最下層のコンピュータまたはプロセッサ同士がローカル・エリア・ネットワークで接続された構成において、通信制御プロセッサを使用せずに任意のグループに分割し、負荷分散を精度よく実施することである。

【0012】

【課題を解決するための手段】 本発明の第1の負荷分散システムは、1以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムであって、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの1つを負荷情報収集コンピュータとし、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第1の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第2の平均負荷情報を収集し、前記第1の平均負荷情報、および第2の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持する前記負荷情報収集コンピュータを有する。

【0013】 本発明の第2の負荷分散システムは、前記第1の負荷分散システムであって、前記自身が属する前記負荷分散グループ内の前記コンピュータの負荷がある一定値を超えた場合に、対応する前記コンピュータの情報を前記負荷情報から削除する前記負荷情報収集コンピュータを有する。

【0014】 本発明の第3の負荷分散システムは、前記

第1または第2の負荷分散システムであって、前記ローカル・エリア・ネットワークに接続され、1つの前記コンピュータと同等のレベルとして扱われる下位の階層の前記ローカル・エリア・ネットワークを有する。

【0015】 本発明の第4の負荷分散システムは、前記第1、第2または第2の負荷分散システムであって、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送する前記負荷情報収集コンピュータを有する。

【0016】 本発明の第1の負荷分散方法は、1以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの1つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第1の平均負荷情報として保持し、自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第2の平均負荷情報を収集し、前記第1の平均負荷情報、および第2の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させる手順を含む。

【0017】 本発明の第2の負荷分散方法は、前記第1の負荷分散方法であって、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含む。

【0018】 本発明の記録媒体は、1以上のコンピュータを接続する複数のローカル・エリア・ネットワークをワイド・エリア・ネットワークまたは、ローカル・エリア・ネットワークで接続した負荷分散システムを使用し、前記コンピュータを任意に負荷分散グループに分割し、それぞれの前記負荷分散グループ内の前記コンピュータの1つを負荷情報収集コンピュータとし、前記負荷情報収集コンピュータに、自身が属する前記負荷分散グループ内の全ての前記コンピュータから負荷を収集し、負荷の低い順位でソートし負荷情報として保持し、前記負荷の平均を算出し第1の平均負荷情報として保持し、

自身が属さない他の前記負荷分散グループの前記負荷情報収集コンピュータから対応する第2の平均負荷情報を収集し、前記第1の平均負荷情報、および第2の平均負荷情報に基づいて、負荷の低い順位でソートしグループ負荷情報として保持させ、かつ、自身が属する前記負荷分散グループ内の前記コンピュータからの処理要求を受け付けると、まず、前記グループ負荷情報によって、最も平均負荷の低い前記負荷分散グループに対して前記処理要求を転送し、もし、最も負荷の低い前記負荷分散グループが自身が属する前記負荷分散グループであれば、最も負荷の低い前記コンピュータに対して前記処理要求を転送させる手順を含むプログラムを記録する。

#### 【0019】

【発明の実施の形態】次に、本発明の第1の実施の形態について図面を参照して詳細に説明する。図1は、本発明の第1の実施の形態を示すブロック図である。図1を参照すると、本発明の第1の実施の形態は、ローカル・エリア・ネットワーク（以降、LANと記す）であるLAN101と、LAN101に接続されるコンピュータ11、コンピュータ12、コンピュータ13、コンピュータ14と、LAN102と、LAN102に接続されるコンピュータ15、コンピュータ16、コンピュータ17、コンピュータ18と、LAN103と、LAN103に接続されるコンピュータ19、コンピュータ20、コンピュータ21、コンピュータ22と、LAN101～103を接続するワイド・エリア・ネットワーク（以降、WANと記す）であるWAN200とから構成される。

【0020】LAN101で接続されているコンピュータ11～コンピュータ14を負荷分散グループとして形成し、これを負荷分散グループG1とする。LAN102で接続されているコンピュータ15～コンピュータ18を負荷分散グループとして形成し、これを負荷分散グループG2とする。LAN103で接続されているコンピュータ19～コンピュータ22を負荷分散グループとして形成し、これを負荷分散グループG3とする。

【0021】負荷分散グループG1内のコンピュータ11は、自グループ内のコンピュータ11～コンピュータ14の負荷情報（たとえば、CPU使用率等）および負荷分散グループG2、負荷分散グループG3の負荷情報を収集する機能と、収集した負荷情報をもとに、他グループの負荷情報を収集するコンピュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ11を負荷分散グループG1の負荷情報収集コンピュータA1とする。

【0022】負荷分散グループG2内のコンピュータ15は自グループ内のコンピュータ15～コンピュータ18の負荷情報および負荷分散グループG1、負荷分散グループG3の負荷情報を収集する機能と、収集した負荷情報をもとに、他グループの負荷情報を収集するコンピ

ュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ15を負荷分散グループG2の負荷情報収集コンピュータA2とする。

【0023】負荷分散グループG3内のコンピュータ19は自グループ内のコンピュータ19～コンピュータ22の負荷情報および負荷分散グループG1、負荷分散グループG2の負荷情報を収集する機能と、収集した負荷情報をもとに他グループの負荷情報を収集するコンピュータまたは自グループの負荷の低いコンピュータに処理要求を転送する機能を有している。このコンピュータ19を負荷分散グループG3の負荷情報収集コンピュータA3とする。

【0024】図1において、コンピュータ11～コンピュータ14は、LAN101に、コンピュータ15～コンピュータ18は、LAN102に、コンピュータ19～コンピュータ22は、LAN103にそれぞれ接続されているが、LAN101、LAN102、LAN103は、LANではなくWANであってもよい。また、各グループはWAN200を介して接続されているが、WAN200ではなくルータ等を介して接続されてもよい。また、各グループは一つのLAN上で構成されているが、一つのLAN上に複数のグループを構成してもよい。また、負荷情報収集コンピュータA1、負荷情報収集コンピュータA2、負荷情報収集コンピュータA3は、負荷分散のための処理とともに、他のコンピュータと同様に通常の処理も実行する。また、負荷情報収集コンピュータは、グループ内のどのコンピュータにも割り当て可能である。

【0025】次に、本発明の第1の実施の形態の動作について図面を参照して説明する。図2～図6は、本発明の第1の実施の形態の動作を示すフローチャートである。図7は、負荷情報および平均負荷情報を示す説明図である。まず、図2を参照して各負荷分散グループ内での負荷情報収集の動作を説明する。

【0026】負荷情報収集コンピュータA1（コンピュータ11）は、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～コンピュータ14）からそれぞれのコンピュータの負荷情報を収集する（図2ステップS1）。

【0027】自分の所属するグループの全てのコンピュータの負荷情報を収集した終えた後（図2ステップS2）、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする（図2ステップS3、図7（a））。ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値（たとえば、CPU使用率80 [%]）を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する（図2ステップS4～S5）。

【0028】次に、図3を参照して、負荷分散グループ



の平均負荷情報収集の動作を説明する。

【0029】負荷情報収集コンピュータA1は、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～14）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する（図3ステップS6～S7、図7（b））。

【0030】さらに他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2（コンピュータ15）、負荷情報収集コンピュータA3（コンピュータ19）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する（図3ステップS8～S10、図7（c））。

【0031】次に、図4を参照して、各負荷分散グループへの平均負荷情報提供の動作を説明する。

【0032】負荷情報収集コンピュータA1は、他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2、負荷情報収集コンピュータA3から平均負荷情報を要求された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータA2、または負荷情報収集コンピュータA3へ自グループ（負荷分散グループG1）の平均負荷情報を転送する（図4ステップS11）。

【0033】次に、図5を参照して、自グループ内のコンピュータから処理要求を受け付けた場合の負荷分散先決定動作を説明する。

【0034】負荷情報収集コンピュータA1は、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～コンピュータ14）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内で最も負荷の低いグループから順に処理要求転送先を決定する（図5ステップS12）。

【0035】最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する（図5ステップS13 YES～S14、たとえば、図7の例では、負荷分散グループG2の負荷情報収集コンピュータA2に対して処理要求を転送する）。

【0036】最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する（図5ステップS13 NO、S15、S16、図7の例では、コンピュータ12に対して処理要求を転送する）。

【0037】次に、図6を参照して、負荷情報収集コンピュータA1が、他グループ（たとえば、負荷分散グループG2）の負荷情報収集コンピュータA2から処理要求を受け付けた場合の負荷情報収集コンピュータA2の

負荷分散動作について説明する。負荷情報収集コンピュータA1は、ソートされた自グループ（負荷分散グループG1）内の負荷情報（図7（a））をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する（図6ステップS17～S18、図7の例では、コンピュータ12に対して処理要求を転送する）。

【0038】次に、本発明の第2の実施の形態について図面を参照して詳細に説明する。図8は、本発明の第2の実施の形態を示すブロック図である。図8を参照すると、本発明の第2の実施の形態は、WAN200を介してLAN102から、さらにLAN103が接続された構成となっている。

【0039】この構成の場合、負荷分散グループG2の負荷情報収集コンピュータA2は、負荷分散グループG3を自グループに属する論理的な1台のコンピュータとして扱う。つまり、負荷情報収集コンピュータA2が自グループの負荷情報を収集する際、負荷分散グループG3の平均負荷情報を自グループ内に存在する一台のコンピュータのものとして収集する。

【0040】また、負荷分散グループG1の負荷情報収集コンピュータA1は、他グループの平均負荷情報として負荷分散グループG2の平均負荷情報のみを収集する。すなわち、直接、負荷分散グループG3の平均負荷情報を収集しない。

【0041】負荷分散グループG2の平均負荷情報は、コンピュータ15～コンピュータ17の負荷情報と負荷分散グループG3の平均負荷情報とをもとに算定される。

【0042】また、負荷分散グループG3の負荷情報収集コンピュータA3は、自グループのコンピュータの負荷情報と、他グループ（負荷分散グループG1、負荷分散グループG2）の平均負荷情報を収集する。

【0043】この方法により、負荷分散グループを階層化することが可能であり、負荷分散グループG2からさらに負荷分散グループG3が後で追加された場合でも、負荷分散グループG1はこの負荷分散グループG3の追加を認識する必要はない。

【0044】次に、本発明の第3の実施の形態について図面を参照して詳細に説明する。図9は、本発明の第3の実施の形態を示すブロック図である。図9を参照すると、本発明の第3の実施の形態は、LAN101とLAN102との間にWAN201、WAN202が接続された構成となっている。

【0045】このような構成で、コンピュータ16は、コンピュータ11～コンピュータ14が接続されているLAN101とは別のLAN102に接続されているが、物理構成とは関係なく同一の負荷分散グループとして構成される。

【0046】次に、本発明の第4の実施の形態について図面を参照して詳細に説明する。図10は、本発明の第

4の実施の形態を示すブロック図である。図10を参照すると、本発明の第4の実施の形態は、接続形態が本発明の第3の実施の形態と同じであるが、コンピュータ15が二つの負荷分散グループG1、負荷分散グループG1に所属している。このように一つのコンピュータが複数の負荷分散グループに所属することも可能である。

【0047】次に、本発明の第5の実施の形態について図面を参照して詳細に説明する。負荷情報収集コンピュータA1への適用例として説明する。本発明の第5の実施の形態は、負荷分散方法である。

【0048】すなわち、本発明の第5の実施の形態は、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～コンピュータ14）からそれぞれのコンピュータの負荷情報を収集させる第1のステップ（図2ステップS1）と、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする第2のステップ（図2ステップS3、図7

(a)）と、ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する第3のステップ（図2ステップS4～S5）と、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～14）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する第4のステップ（図3ステップS6～S7、図7（b)）と、他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2（コンピュータ15）、負荷情報収集コンピュータA3（コンピュータ19）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する第5のステップ（図3ステップS8～S10、図7（c)）と、他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2、負荷情報収集コンピュータA3から平均負荷情報を要求された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータA2、または負荷情報収集コンピュータA3へ自グループ（負荷分散グループG1）の平均負荷情報を転送する第6のステップ（図4ステップS11）と、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～コンピュータ14）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内でも最も負荷の低いグループから順に処理要求転送先を決定する第7のステップ（図5ステップS12）と、最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する第8のステップ（図5ステップS13YES～S14、たとえば、図7の例では、負荷分散グループG2の負荷情報収集コンピュータA2に対して処理要

求を転送する）と、最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する第9のステップ（図5ステップS13NO、S15、S16、図7の例では、コンピュータ12に対して処理要求を転送する）と、ソートされた自グループ（負荷分散グループG1）内の負荷情報（図7（a)）をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する第10のステップ（図6ステップS17～S18、図7の例では、コンピュータ12に対して処理要求を転送する）とを含む。

【0049】次に、本発明の第6の実施の形態について図面を参照して詳細に説明する。図11は、本発明の第6の実施の形態を示すブロック図である。図11を参照すると、本発明の第6の実施の形態は、本発明の第5の実施の形態の負荷分散方法をコンピュータ11～コンピュータ22に実行させるプログラムを記録した記録媒体300である。この記録媒体300は磁気ディスク、半導体メモリ、その他であってよい。負荷分散方法を記録したプログラムは記録媒体300から各コンピュータ11～コンピュータ22に読み込まれ、各コンピュータの負荷分散動作を制御する。各コンピュータはこの読み込まれた負荷分散方法プログラムによって本発明の第1の実施の形態で説明した処理と同一の処理を行う。

【0050】詳細には、本発明の第6の実施の形態は、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～コンピュータ14）からそれぞれのコンピュータの負荷情報を収集させる第1のステップ（図2ステップS1）と、収集した自グループの全てのコンピュータの負荷情報を負荷の低い順にソートする第2のステップ（図2ステップS3、図7（a)）と、ソートが完了した場合、あらかじめ環境設定等により、指定された負荷のしきい値を超えているコンピュータがあれば、そのコンピュータの負荷情報は削除する第3のステップ（図2ステップS4～S5）と、自分の所属するグループ（負荷分散グループG1）の各コンピュータ（コンピュータ11～14）の負荷の値の平均を算出し、自グループの平均負荷情報として自グループの負荷情報とは別に保持する第4のステップ（図3ステップS6～S7、図7（b)）と、他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2（コンピュータ15）、負荷情報収集コンピュータA3（コンピュータ19）へそれぞれのグループの平均負荷情報の転送を要求し、自グループの平均負荷情報とともに負荷の低い順にソートし保持する第5のステップ（図3ステップS8～S10、図7（c)）と、他グループ（負荷分散グループG2、負荷分散グループG3）の負荷情報収集コンピュータA2、負荷情報収集コンピュータA3から平均負荷情報を要求

された時のみ、それぞれ要求された他グループの負荷情報収集コンピュータ A 2、または負荷情報収集コンピュータ A 3 へ自グループ（負荷分散グループ G 1）の平均負荷情報を転送する第 6 のステップ（図 4 ステップ S 1 1）と、自分の所属するグループ（負荷分散グループ G 1）の各コンピュータ（コンピュータ 1 1 ～コンピュータ 1 4）から処理要求を受け付けた場合、まずソートされた各負荷分散グループの平均負荷情報をもとに各グループの内で最も負荷の低いグループから順に処理要求転送先を決定する第 7 のステップ（図 5 ステップ S 1 2）と、最も負荷の低いグループが他グループであった場合、そのグループの負荷情報収集コンピュータに処理要求を転送する第 8 のステップ（図 5 ステップ S 1 3 Y E S ～ S 1 4、たとえば、図 7 の例では、負荷分散グループ G 2 の負荷情報収集コンピュータ A 2 に対して処理要求を転送する）と、最も負荷の低いグループが自グループであった場合、さらにソートされた自グループ内の負荷情報をもとに自グループ内でもっとも負荷の低いコンピュータから順に、処理要求を転送する第 9 のステップ（図 5 ステップ S 1 3 N O、S 1 5、S 1 6、図 7 の例では、コンピュータ 1 2 に対して処理要求を転送する）と、ソートされた自グループ（負荷分散グループ G 1）内の負荷情報（図 7（a））をもとに自グループ内で最も負荷の低いコンピュータから順に処理要求を転送する第 10 のステップ（図 6 ステップ S 1 7 ～ S 1 8、図 7 の例では、コンピュータ 1 2 に対して処理要求を転送する）とをコンピュータに実行させるプログラムを記録した記録媒体である。

#### 【0051】

【発明の効果】本発明の第 1 の効果は、負荷情報の収集処理を行うことによる CPU 負荷を複数台のコンピュータに分散させることができることである。

【0052】その理由は、各コンピュータをグループ化し、各グループの代表のコンピュータは自グループ内の各コンピュータの負荷情報と他グループの平均負荷情報とを収集することで、負荷情報の収集処理を分散させることが可能となり、負荷情報の収集処理を行うことによる CPU 負荷を複数台のコンピュータに分散させるからである。

【0053】第 2 の効果は、1 台のコンピュータに対する負荷情報を保持するメモリ（または記録媒体）量を押さえることができることである。

【0054】その理由は、各コンピュータをグループ化し、各グループの代表のコンピュータが負荷情報収集処理を行うので、システム内の全コンピュータの負荷情報を 1 台のコンピュータで保持する必要がないからである。

【0055】第 3 の効果は、各グループ間が WAN 等、従量制で課金されるようなネットワークで接続されていた場合に、通信費用を押さえることができることであ

る。

【0056】その理由は、グループ毎に負荷情報を平均化し、他グループの代表のコンピュータはこの平均化された負荷情報のみ収集すればよいため通信データ量が少なくてすむからである。

【0057】第 4 の効果は、各システムの環境設定を局所化することが可能となり、システム構築の柔軟性が高くなることである。

【0058】その理由は、各グループの負荷情報を収集するコンピュータは、自グループ以外のコンピュータは管理しないため、別グループにコンピュータが新たに設置または撤去された場合、そのコンピュータの設置または撤去における各コンピュータへの環境設定等への影響は当該グループのみとなるからである。

【0059】第 5 の効果は、コンピュータの障害を局所化することが可能となることである。

【0060】その理由は、各グループの負荷情報を収集するコンピュータは、自グループ以外のコンピュータは管理しないため、各グループの負荷情報を収集するコンピュータ以外のコンピュータに障害が発生したとしても、その障害を検知する必要がないからである。

【0061】また、負荷情報を収集するコンピュータが他グループの平均負荷情報の収集を停止すれば他グループから処理要求の転送が行われなくなるため、平均負荷情報の収集を停止したグループを負荷分散の対象から切り離すことが可能となり、これにより定期保守等をシステム全体に影響を及ぼすことなく行うことが可能となるからである。

【0062】第 6 の効果は、構成が簡単なことである。

【0063】その理由は、通信制御プロセッサを必要としないからである。

#### 【図面の簡単な説明】

【図 1】本発明の第 1 の実施の形態を示すブロック図である。

【図 2】本発明の実施の形態の動作を示すフローチャートである。

【図 3】本発明の実施の形態の動作を示すフローチャートである。

【図 4】本発明の実施の形態の動作を示すフローチャートである。

【図 5】本発明の実施の形態の動作を示すフローチャートである。

【図 6】本発明の実施の形態の動作を示すフローチャートである。

【図 7】負荷情報および平均負荷情報を示す説明図である。

【図 8】本発明の第 2 の実施の形態を示すブロック図である。

【図 9】本発明の第 3 の実施の形態を示すブロック図である。

【図10】本発明の第4の実施の形態を示すブロック図である。

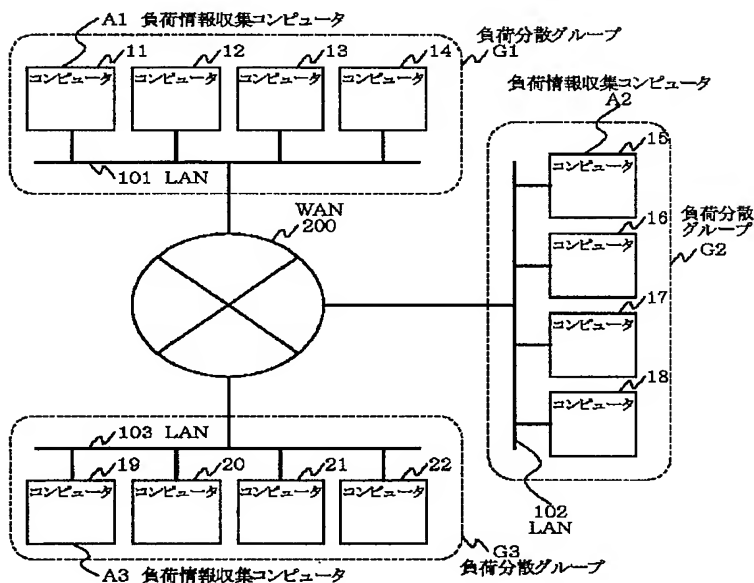
【図11】本発明の第6の実施の形態を示すブロック図である。

【符号の説明】

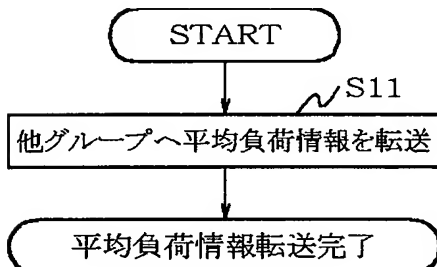
11 コンピュータ  
12 コンピュータ  
13 コンピュータ  
14 コンピュータ  
15 コンピュータ  
16 コンピュータ  
17 コンピュータ  
18 コンピュータ  
19 コンピュータ  
20 コンピュータ

21 コンピュータ  
22 コンピュータ  
101 LAN  
102 LAN  
103 LAN  
200 WAN  
201 WAN  
202 WAN  
300 記録媒体  
A1 負荷情報収集コンピュータ  
A2 負荷情報収集コンピュータ  
A3 負荷情報収集コンピュータ  
G1 負荷分散グループ  
G2 負荷分散グループ  
G3 負荷分散グループ

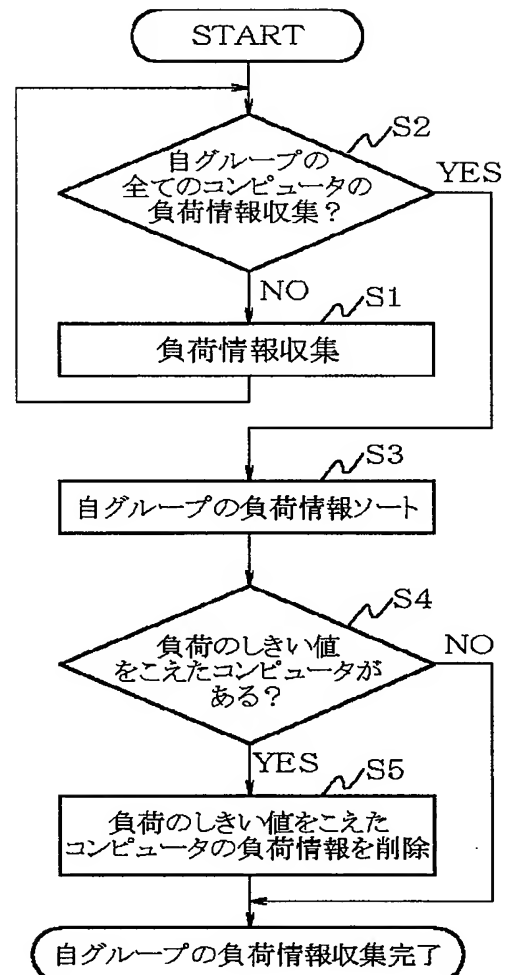
【図1】



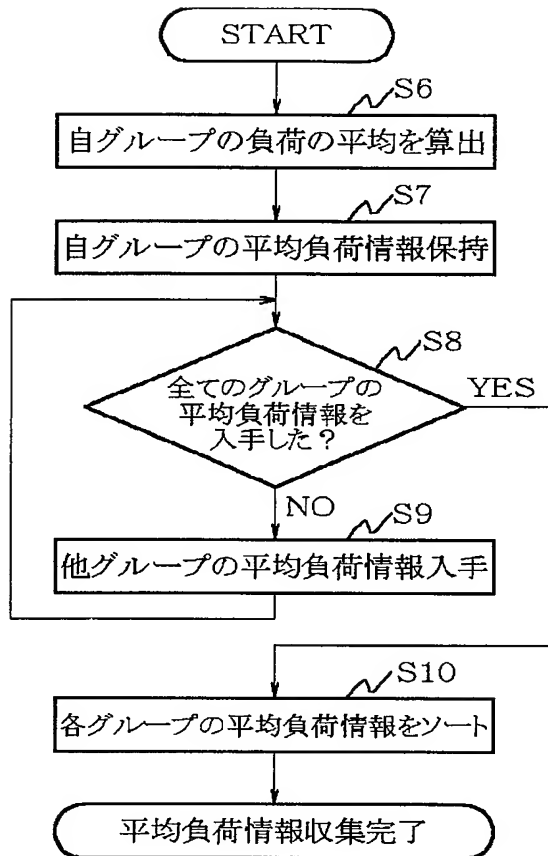
【図4】



【図2】



【図3】



【図7】

(a)

順位	CPU使用率
コンピュータ12	30
コンピュータ11	40
コンピュータ14	50
コンピュータ13	70

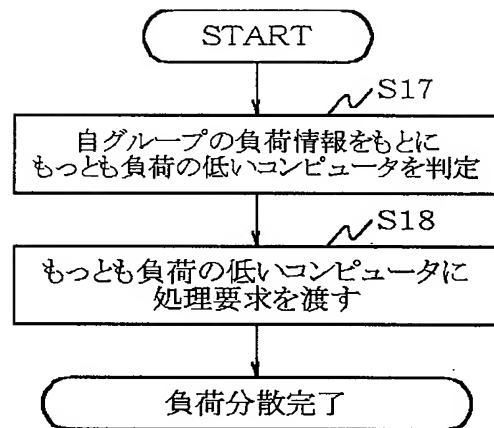
(b)

平均負荷
$(30+40+50+70)/4=47.5$

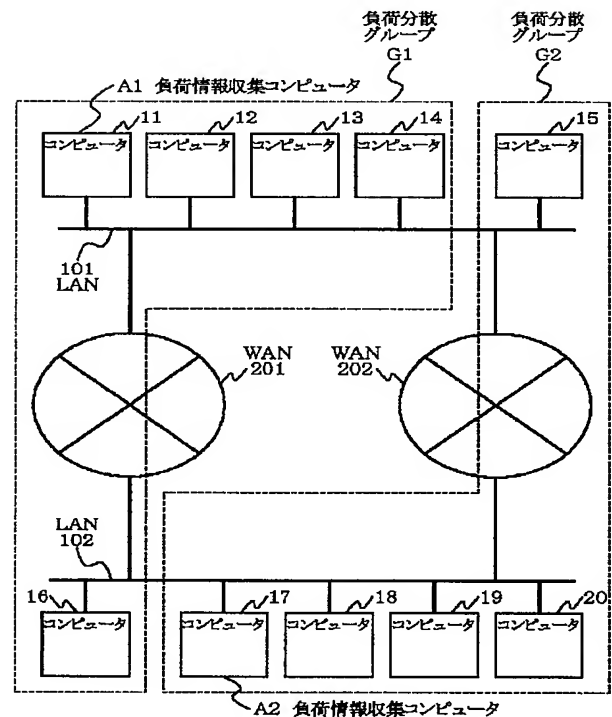
(c)

順位	平均負荷
負荷分散グループG2	30.0
負荷分散グループG1	47.5
負荷分散グループG3	50.0

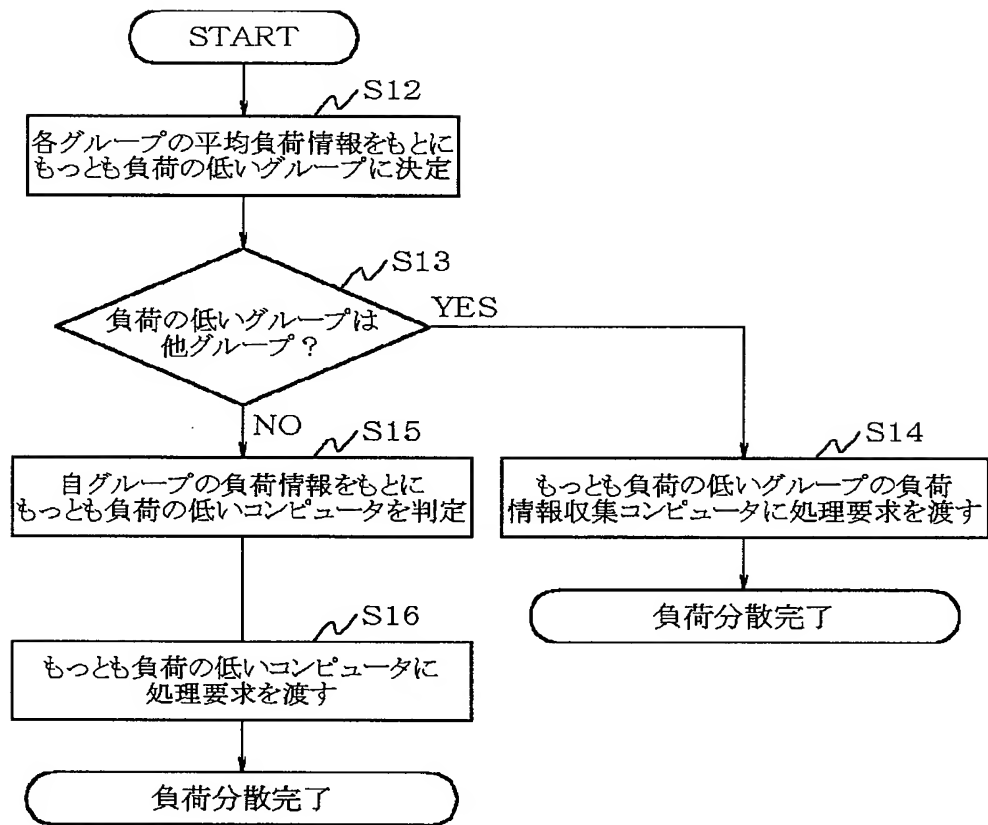
【図6】



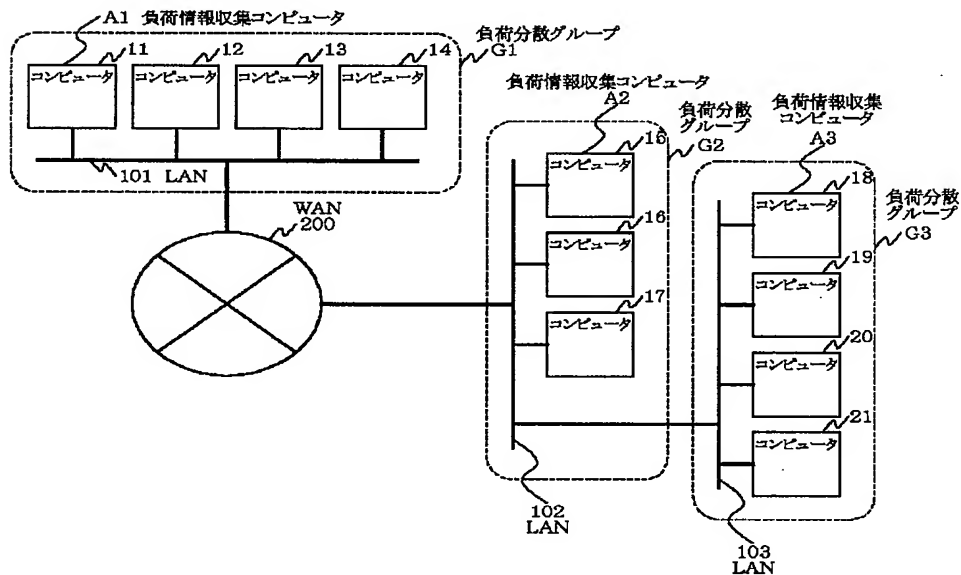
【図9】



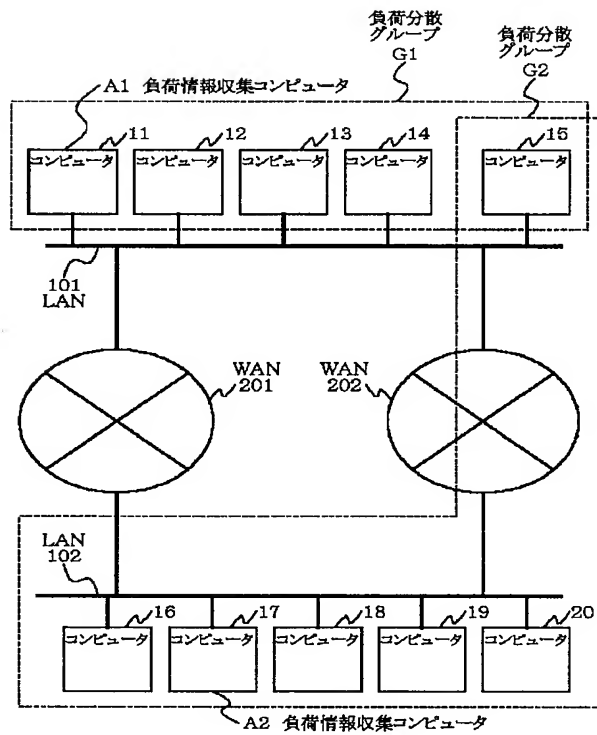
【図 5】



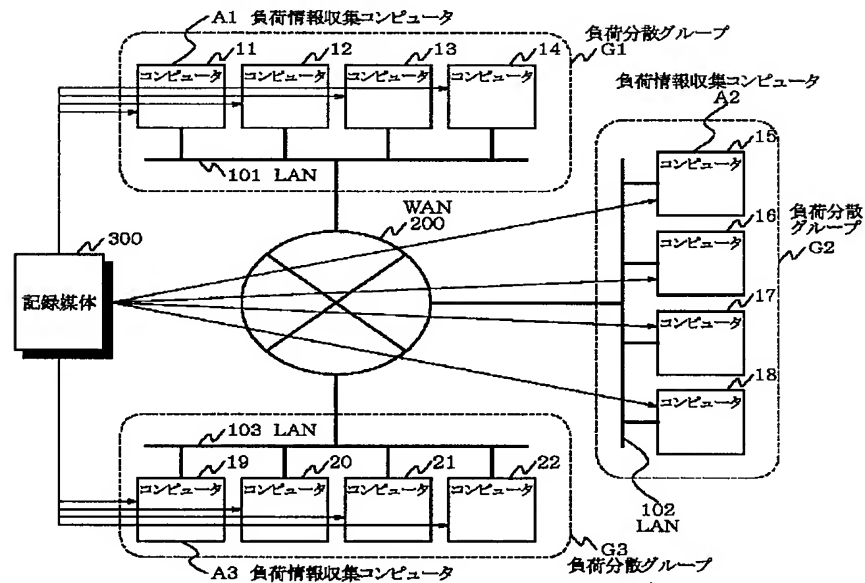
【図 8】



【図 10】



【図 11】



# PATENT ABSTRACTS OF JAPAN

(11)Publication number : **2000-148783**

(43)Date of publication of application : **30.05.2000**

(51)Int.Cl. **G06F 17/30**

**G06F 13/00**

(21)Application number : **10-324249**

(71)Applicant : **NEC CORP**

(22)Date of filing : **13.11.1998**

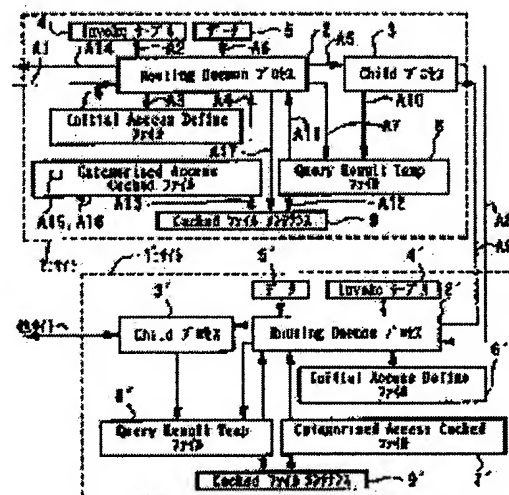
(72)Inventor : **KIKUCHI SHINJI**

## (54) DATA RETRIEVAL DEVICE

### (57)Abstract:

**PROBLEM TO BE SOLVED:** To widen the area of data retrieval using the internet and to make the retrieval efficient.

**SOLUTION:** When a site 1 receives a data retrieval request A1 from a user, a Routing Daemon process 2 actuates a Child process 3 by referring to data 5 in its device and makes another side 1' outputs a data retrieval request A8. This data retrieval request A8 includes address information A3 on a site to which the data retrieval request is always transferred and address information A4 on a site to which the most likelihood data retrieval request should be transferred from an Initial Access Define file 6. Transfer data A9 retrieved at the other side 1' are held in a Query Result Temp file 8 together with corresponding data A7 by the data 5, merged, and sent as transfer data A14 back to the user. Cached file maintenance 9 extracts the address information on the most likelihood site according to data contents transferred from the Query Result Temp file 8 and updates the Query Result Temp file 8.





(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-148783

(P2000-148783A)

(43) 公開日 平成12年5月30日 (2000.5.30)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	テーマコード* (参考)
G 0 6 F 17/30		G 0 6 F 15/40	3 1 0 F 5 B 0 7 5
13/00	3 5 4	13/00	3 5 4 D 5 B 0 8 9
		15/40	3 1 0 C

審査請求 有 請求項の数11 O L (全 10 頁)

(21) 出願番号 特願平10-324249

(22) 出願日 平成10年11月13日 (1998. 11. 13)

(71) 出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72) 発明者 菊地 伸治

東京都港区芝五丁目7番1号 日本電気株式会社内

(74) 代理人 100108578

弁理士 高橋 昭男 (外3名)

Fターム(参考) 5B075 KK02 KK13 KK33 PQ05 PQ20

5B089 GA11 GB09 HA10 JA12 KC15

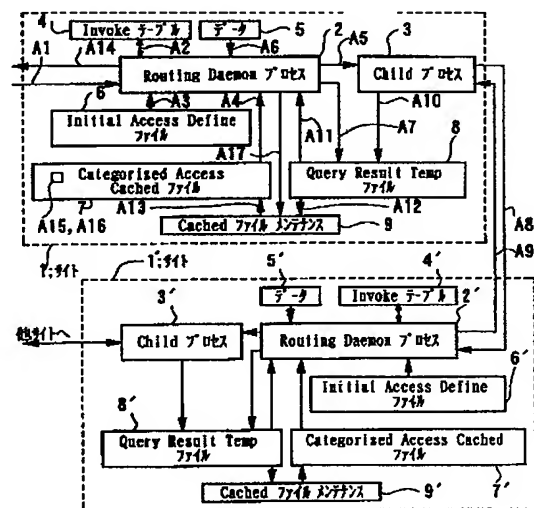
KC23 KC39 KC44

## (54) 【発明の名称】 データ検索装置

## (57) 【要約】

【課題】 インターネットを利用したデータ検索の広域化、高効率化を図る。

【解決手段】 サイト1が利用者からデータ検索要求A1を受信すると、RoutingDaemonプロセス2は、自装置内のデータ5を参照し、Childプロセス3を起動し、他のサイト1'にもデータ検索要求A8を出力させる。このデータ検索要求A8には、InitialAccessDefineファイル6からの、常にデータ検索要求を転送する サイトのアドレス情報A3と、最も確からしいデータ検索要求を転送すべきサイトのアドレス情報A4を含む。他のサイト1'で検索された転送データA9は、データ5による該当データA7と共にQueryResultTempファイル8に保持され、マージされ、転送データA14として利用者に返される。Cachedファイルメンテナンス9は、QueryResultTemp ファイル8から転送されて きたデータ内容に応じて最も確からしいサイトのアドレス情報を引き出し、QueryResultTemp ファイル8を更新する。



A1: データ検索要求  
A2: 情報  
A3: IPアドレス情報  
A4: IPアドレス情報  
A5: 引数  
A6: サブセット  
A7: 該当データ  
A8: データ検索要求  
A9: 転送データ  
A10: 該当データ

A11: 該当データ  
A12: パラメータ  
A13: 更新元データ  
A14: 転送データ  
A15: 相対網番値  
A16: 相対網番値  
A17: 起動要求

**【特許請求の範囲】**

**【請求項 1】** インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、

データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置からデータを受信し、前記自装置内に保持しているデータとマージしてデータ検索要求元に転送することを特徴とするデータ検索装置。

**【請求項 2】** インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、

データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置から検索データを受信する常駐の Routing Daemon プロセスと、  
該 Routing Daemon プロセスから起動され、複数の他のデータ転送装置とデータ転送に関する通信を行なう Child プロセスとを有することを特徴とするデータ検索装置。

**【請求項 3】** 前記各データ転送装置は、新たなデータ検索要求を受けると、前記 Routing Daemon プロセスによって起動識別子が登録される Invoke テーブルを備え、Routing Daemon プロセスは、データ検索要求を受けた場合に、該 Invoke テーブルを参照することにより、重複した処理を回避することを特徴とする請求項 2 記載のデータ検索装置。

**【請求項 4】** 前記各データ転送装置は、前記 Routing Daemon プロセスが、常に前記データ検索要求を転送する他のデータ転送装置のアドレスを格納した Initial Access Define ファイルを備えたことを特徴とする請求項 2 または請求項 3 記載のデータ検索装置。

**【請求項 5】** 前記各データ転送装置は、前記 Routing Daemon プロセスが、前記データ検索要求のデータ内容に応じて最も確からしいデータ検索要求を転送すべきデータ転送装置のアドレスを格納した Categorized Access Cached ファイルを備えたことを特徴とする請求項 2 ないし請求項 4 のいずれかに記載のデータ検索装置。

**【請求項 6】** 前記各データ転送装置は、前記データ検索の結果、前記自装置内で参照されたデータおよび他のデータ転送装置から転送された検索データを一時的に保持する Query Result Temp ファイルと、該 Query Result Temp ファイルから転送されてきたデータ内容に応じて最も確からしいデータ転送先のデータ転送装置のアドレスを引き出し、この結果によって前記 Categorized Access Cached ファイルを更新する Cached ファイル メンテナンスとを備えたことを特徴とする請求項 5 記載のデータ検索装置。

**【請求項 7】** 前記最も確からしいデータ転送装置は、まず、検索条件の指定カテゴリに対し相対網羅度が最小のものを選択し、次に、相対新規度が最小のものを第 1 優先条件、前記相対網羅度が最小のものを第 2 優先条件としてソートして決定することを特徴とする請求項 5 または請求項 6 記載のデータ検索装置。

**【請求項 8】** 前記最も確からしいデータ転送装置の決定方法は、計算上、扱うべきデータ転送装置の数を制限するサイト上限値設定手順と、保持している検索データ件数が多い順にソートして前記制限されたデータ転送装置の数に相当する順位迄のデータ転送装置を選択する該当サイト選択手順と、検索結果の対象をソートし、同じ対象に関する記述を持つ異なるデータ転送装置の数を前記対象の各々に対して数える該当オブジェクト選択手順と、前記選択された全データ転送装置に対して、各々、データ転送装置の数を変数 X とし、自身を含めた X 台のデータ転送装置に含まれる前記対象の度数に関するヒストグラムを作成するヒストグラム作成手順と、該作成されたヒストグラムを基に網羅度および新規度を計算する網羅度・新規度算出手順と、前記網羅度の小さい順にソートしてデータ転送装置を並び替え、小さい順に前記相対網羅度を付与する相対網羅度付与手順と、前記新規度の小さい順にソートしてデータ転送装置を並び替え、小さい順に前記相対度を付与する相対新規度付与手順と、該付与された相対網羅度、相対新規度と、前記 Categorized Access Cached ファイルで管理されている現在の相対網羅度、相対新規度との相対平均をとり、前記 Categorized Access Cached ファイルに書き込む更新手順とを含むことを特徴とする請求項 7 記載のデータ検索装置。

**【請求項 9】** 前記 Child プロセスは、起動の際、データ検索要求を転送すべき他のデータ転送装置のアドレス情報および他のデータ転送装置から転送されてきたデータを一時的に保持すべき前記 Query Result Temp ファイルの名称情報を前記 Routing Daemon プロセスから引数として受け取ることを特徴とする請求項 2 ないし請求項 8 のいずれかに記載のデータ検索装置。

**【請求項 10】** 前記データ検索要求は、転送される際に、通過するデータ転送装置の Name 特定情報および累積 Hop 数が内部にシーケンス状に記録されることを特徴とする請求項 1 ないし請求項 9 のいずれかに記載のデータ検索装置。

**【請求項 11】** 前記データ検索要求、Invoke テーブル、Initial Access Define ファイル、Categorized Access Cached ファイルおよび Query Result Temp ファイル BNF (Backus Normal Form) で定義される記述形式で表されることを特徴とする請求項 1 ないし請求項 10 のいずれかに記載のデータ検索装置。

**【発明の詳細な説明】**

**【0001】**

**【発明の属する技術分野】** 本発明は、インターネット上

のデータ検索要求に対するデータ転送装置に関する。

#### 【0002】

【従来の技術】従来のこの種の転送方式を図4に示す。この方式は、米国論文 (M. A. sheldon, A. Duda, R. Weiss, D. K. Gifford: Discover: A Resource Discovery System based on Content Routing, The 3<sup>rd</sup> International world-wide web Conference April 10-14, 1995 (<http://www.igd.de/www/www95/papers/82/html-files/discover.htm> 1) (1995).) に記されたものである。

【0003】この従来方式は、利用者に対してサーバの役割を果たす Httpd プロセス 11, ルーティング機能を提供する複数のコンテンツルータ 12 等, 各コンテンツルータが参照し、ルーティングに関する情報を管理する複数のルーティングデータベース 13 等, 各コンテンツルータが参照し、利用者に正しい質問入力を促進するための複数の Refinement データベース 14 等, 検索機能を提供する複数のサーチモジュール 16, ドキュメントデータを管理するためのシステムである複数の WAIS プロセス 17 および各 WAIS プロセス 17 が参照する複数の WAIS データベース 18 を含んでいる。

【0004】利用者が Web ブラウザ等を介して、Httpd プロセス 11 にドキュメントの検索要求 B1 を発行すると、Httpd プロセス 11 は、ルーティング機能を提供し、直接関連しているコンテンツルータ 12 に検索要求 B12 を発行する。このコンテンツルータ 12 は、通常、Httpd プロセス 11 のバックプロセスとして機能する。

【0005】その後、コンテンツルータ 12 は、検索要求 B1 で指定され、検索要求 B2 に含まれる質問語彙妥当か否かを判定するため、Refinement データベース 14 上の正しい質問データ B3 を参照し、妥当な表現に修正する。

【0006】次に、コンテンツルータ 12 は、ルーティングデータベース 13 に問い合わせ、質問語彙に対応して、該当する検索結果を応答できる、もしくはその仲立ちができる他のコンテンツルータ 12', 12'' の位置情報 B4 を得る。この情報はルーティングデータベース 13 の内部ではコンテンツラベル 15 として管理されている。また他のルーティングデータベース 13', 13'' でも同様にコンテンツラベル 15', 15'' が存在している。

【0007】次に、コンテンツルータ 12 は、コンテンツラベル 15 から得た位置情報 B4 が指し示す他のコンテンツルータ 12', 12'' に、検索要求 B2 と等価な検索要求 B5, B5' を転送する。

【0008】他のコンテンツルータ 12', 12'' においても、コンテンツルータ 12 と全く同様の処理を行なう。その際、もしこのコンテンツルータ 12', 12'' がルーティングの際に、最終位置にある場合、直接関連するサーチモジュール 16, 16' に検索要求 B5, B

5' と等価な検索要求 B6, B6' を転送する。サーチモジュール 16, 16' が検索要求 B6, B6' を受けると、この検索要求 B6, B6' と等価な検索要求 B7, B7' を配下のドキュメントデータを管理する WAIS プロセス 17, 17' に発行する。それぞれの WAIS プロセス 17, 17' は検索要求 B1 で指定された質問語彙を含む B9, B9' を各 WAIS データベース 18, 18' に発行し、関連するドキュメントの位置情報 B10, B10' を WAIS プロセス 17, 17' に返す。WAIS プロセス 17, 17' は、対応するサーチモジュール 16, 16' に検索要求 B7, B7' の応答として、ドキュメントの位置情報 B10, B10' と等価な位置情報 B11, B11' を返す。

【0009】同様にサーチモジュール 16, 16' は対応するコンテンツルータ 12', 12'' に検索要求 B6, B6' の応答として、ドキュメントの位置情報 B11, B11' と等価な位置情報 B12, B12' を返す。その後コンテンツルータ 12', 12'' は、起点となるコンテンツルータ 12 に検索要求 B5, B5' の応答として、ドキュメントの位置情報 B12, B12' と等価な位置情報 B13, B13' を返す。

【0010】起点となるコンテンツルータ 12 は、配下のコンテンツルータ 12', 12'' から検索結果であるドキュメントの位置情報 B13, B13' を全て受けると、それらを合成し最終的な位置情報 B14 として Httpd プロセス 11 に戻す。その後、利用者が発行したドキュメントの検索要求 B1 に対応したドキュメントの位置情報 B15 が利用者の Web ブラウザ等上に表示されることになる。

【0011】コンテンツラベル 15, 15', 15'' を更新する場合は、コンテンツルータ 12, 12', 12'' は検索条件を限定しない検索要求 B16, B16' を発行する。その後、前述の手順に応じて、WAIS データベース 18, 18' 上に管理された全ドキュメントの位置情報 B17, B17' が返されることになる。全ドキュメントの位置情報 B17, B17' の内容はコンテンツラベル 15, 15', 15'' を更新し得るだけの内容を持ち、情報内容を持って、コンテンツルータ 12, 12', 12'' はコンテンツラベル 15, 15', 15'' を更新する。

#### 【0012】

【発明が解決しようとする課題】上述した従来の方式 2'' では、図4に示した様に WAIS プロセス 17, 17' を使っているため、ドキュメント検索の際、語彙検索が中心となる。また、基本的に WAIS データベース 18, 18' で管理されるコンテンツデータを対象としており、一般的なレガシーデータを対象とはしていないという第1の問題点がある。

【0013】また、コンテンツラベル 15, 15', 15'' を更新する手段を用意しているが、検索の際には

最適化した検索が行われる機構が特になく、決められたコンテンツルータ 12' , 12" にのみ、常に検索要求をルーティングすることしか出来ないという第2の問題点がある。

【0014】また、仮に、2つのルーティングデータベース 13' , 13" 内の各々のコンテンツラベル 15' , 15" 上に、互いに相手の位置情報が登録されている場合には、検索が2重に実施されることになり検索上のロスが発生するという第3の問題点がある。

【0015】従って、本発明の目的は、近年、益々、重要になって来ているインターネット上のドキュメントを含むデータベースの検索を出来るだけ広域にかつ無駄なく、実施出来るデータ転送装置を提供することにある。

【0016】また、本発明の他の目的は、利用者の欲している情報をできるだけ定量的に評価して効率を高めることができるデータ検索装置を提供することにある。

【0017】

【課題を解決するための手段】そこで、本発明の第1のデータ検索装置は、インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置からデータを受理し、前記自装置内に保持しているデータとマージしてデータ検索要求元に転送することを特徴とする。また、本発明の第2のデータ検索装置は、インターネットを利用したデータ検索装置において、複数のデータ転送装置から成り、各データ転送装置は、データ検索要求を受信すると、自装置内に保持しているデータを参照するとともに、所定数に達するまで、他のデータ転送装置へ前記データ検索要求を転送して当該他のデータ転送装置から検索データを受理する常駐のRouting Daemon プロセスと、該Routing Daemon プロセスから起動され、複数の他のデータ転送装置とデータ転送に関する通信を行なうChildプロセスとを有することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、新たなデータ検索要求を受けると、前記Routing Daemon プロセスによって起動識別子が登録されるInvoke テーブルを備え、Routing Daemon プロセスは、データ検索要求を受けた場合に、該Invoke テーブルを参照することにより、重複した処理を回避することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記Routing Daemon プロセスが、常に前記データ検索要求を転送する他のデータ転送装置のアドレスを格納したInitial Access Define ファイルを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記Routing Daemon プロセスが、前記データ検索要求のデータ内容に

応じて最も確からしいデータ検索要求を転送すべきデータ転送装置のアドレスを格納したCategorized Access Cached ファイルを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記各データ転送装置は、前記データ検索の結果、前記自装置内で参照されたデータおよび他のデータ転送装置から転送された検索データを一時的に保持するQuery Result Temp ファイルと、該Query Result Temp ファイルから転送されてきたデータ内容に応じて最も確からしいデータ転送先のデータ転送装置のアドレスを引き出し、この結果によって前記Categorized Access Cached ファイルを更新するCached ファイル メンテナンスとを備えたことを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記最も確からしいデータ転送装置は、まず、検索条件の指定カテゴリに対し相対網羅度が最小のものを選択し、次に、相対新規度が最小のものを第1優先条件、前記相対網羅度が最小のものを第2優先条件としてソートして決定することを特徴とする。さらに、本発明のデータ検索装置の好ましい実施の形態は、前記最も確からしいデータ転送装置の決定方法は、計算上、扱うべきデータ転送装置の数を制限するサイト上限値設定手順と、保持している検索データ件数が多い順にソートして前記制限されたデータ転送装置の数に相当する順位迄のデータ転送装置を選択する該当サイト選択手順と、検索結果の対象をソートし、同じ対象に関する記述を持つ異なるデータ転送装置の数を前記対象の各々に対して数える該当オブジェクト選択手順と、前記選択された全データ転送装置に対して、各々、データ転送装置の数を変数Xとし、自身を含めたX台のデータ転送装置に含まれる前記対象の度数に関するヒストグラムを作成するヒストグラム作成手順と、該作成されたヒストグラムを基に網羅度および新規度を計算する網羅度・新規度算出手順と、前記網羅度の小さい順にソートしてデータ転送装置を並べ替え、小さい順に前記相対網羅度を付与する相対網羅度付与手順と、前記新規度の小さい順にソートしてデータ転送装置を並べ替え、小さい順に前記相対新規度を付与する相対新規度付与手順と、該付与された相対網羅度、相対新規度と、前記Categorized Access Cached ファイルで管理されている現在の相対網羅度、相対新規度との相対平均をとり、前記Categorized Access Cached ファイルに書き込む更新手順とを含むことを特徴とする。

【0018】

【発明の実施の形態】次に、本発明の実施の形態について説明する。図1は、本発明のデータ内容に応じたルーティング方式を採用したデータ転送装置の実施例を示す。本実施例では、データ転送装置をサイト(Site)と称し、図1には2つのサイト1と1' が示されている。

【0019】サイト1にはデータ検索要求を受けると、自装置内に保持しているデータを参照すると共に他サイ

ト 1' へ同一のデータ検索要求を転送・発行し、その結果、他サイト 1' 内に保持しているデータを受理する常駐プログラムの Routing Daemon プロセス 2 と、Routing Daemon プロセス 2 から起動され、他の、サイト 1' 等の複数のデータ転送装置と実際のデータ転送に関する通信を行なう Child プロセス 3 と、データ検索要求の処理状況を管理・記録する Invoke テーブル 4 と、サイト内で固有に保持されているデータ 5 と、Routing Daemon プロセス 2 が常にデータ検索要求を転送・発行するサイトのアドレスを記した Initial Access Define ファイル 6 と、後述の該当データマイニング方式により更新され、データ検索要求を転送・発行する際に最も確からしいサイトのアドレスを得るのに使用する Categorized Access Cached ファイル 7 と、データ検索要求によるデータ 5 への参照結果であるデータ、並びに上述のデータ検索要求の結果、転送されて来たサイト 1' 内に保持されているデータ 5' を一時的に保持・記録する Query Result Temp ファイル 8 と、データマイニング方式により、Query Result Temp ファイル 8 からデータ検索要求の条件に応じて最も確からしいデータ転送先のサイトのアドレスを引き出し、その結果をもって Categorized Access Cached ファイル 7 を更新・維持管理する Cached ファイ

ル メンテナンス 9 を含んでいる。

【0020】Invoke テーブル 4 は、各サイトに 1 つ設定されるものであり、サイト間でやりとりされるデータ検索要求の処理状況を管理・記録するものである。Routing Daemon プロセス 2 がデータ検索要求  $A_1$  を受理すると、Invoke テーブル 4 をアクセスし、該当する起動識別子（以下 InvokeID）を含んだ情報  $A_2$  が存在するか否かを確認する。この InvokeID を含んだ情報  $A_2$  を初めて扱う場合は、InvokeID を含んだ情報  $A_2$  をテーブル 4 に書き込むことで登録する。それに対して、サイト 1 がサイト 1' 内の Child 3' から、既に同一の InvokeID に相当するデータ検索要求  $A_1$  を受けている場合は、その InvokeID を含んだ情報は、登録済みになっているので、新たに同じデータ検索要求を受けた場合、Routing Daemon プロセス 2 が処理を無視する、もしくは拒否する等を行ない、2 重に同一の処理が実施されることを防止する。

【0021】Invoke テーブル 4 は、以下の BNF (Backus Normal Form) で定義される記述形式を持ち、同一のデータ検索要求は容易に判定出来る。

【0022】

【数 1】

```
InvokeTable ::= <originl_pass> <invokeID>;
<originl_pass> ::= <identifier>;
<identifier> ::= <ip_string> | <dns_string> | <other>;
<ip_string> ::= {<number>3} { '.' <number>3 }3;
<dns_string> ::= {<word> '.' } {<word> '.' } *;
<other> ::= <word> *;
<string> ::= <character> | <character> <character> *;
<word> ::= <alphabetic> {<alphabetic> } *;
<character> ::= <alphabetic> | <number> | <special>;
<number> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
<special> ::= <|> | = | + | - | * | / | & | ^ | ~ | _ | @ | $ | % | : | . | ! | ?;
<invokeID> ::= <number>15;
```

【0023】また、データ検索要求  $A_1$  は、以下の BNF で定義される記述形式を持ち、転送の際に通過サイトの Name 特定情報、並びに累積 Hop 数が内部に記憶される。尚、通過サイトの Name 特定情報は、順序が把握出

来るようにシーケンス状に記録される。

【0024】

【数 2】

```

QueryText ::= (originl_pass) (invokeID) (query) (hop_number) (pass_list) (local_pass);
(hop_number) ::= (number) 3;
(pass_list) ::= ' | ' { (remote_pass) ' | ' } *;
(remote_pass) ::= (identifier);
(local_pass) ::= (identifier);
(originl_pass) ::= (identifier);
(identifier) ::= (ip_string) | (dns_string) | (other);
(ip_string) ::= { (number) 3 } { ' . ' (number) 3 } 3;
(dns_string) ::= { (word) ' . ' } { (word) ' . ' } *;
(other) ::= (word) *;
(query) ::= (string);
(string) ::= (character) | (character) (character) *;
(word) ::= (alphabetic) { (alphabetic) } *;
(character) ::= (alphabetic) | (number) (special);
(number) ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
(special) ::= ( | ) | = | + | - | * | / | & | ^ | ~ | _ | @ | $ | % | : | . | ! | ?;
(invokeID) ::= (number) 15;

```

【0025】 Routing Daemon プロセス2は、その後、categorized Access Cached ファイル7をアクセスし、先のデータ検索要求A<sub>1</sub>を転送・発行するべきサイトのアドレス情報A<sub>4</sub>を得ると共に、Initial Access Define ファイル6もアクセスし、同様に先のデータ検索要求A<sub>1</sub>を転送・発行するサイトのアドレス情報A<sub>3</sub>を得る。Initial Access Define ファイル6には、Routing Daemon プロセス2が常にデータ検索要求A<sub>1</sub>を転送・発行するサイトのアドレスが記されているのに対して、Categorized Access Cached ファイル7には、データ検索要求A<sub>1</sub>で指定された検索条件としてのカテゴリに応じて、データ検索要求を転送・発行する際に最も確からしいサイトのアドレスが記されている。

【0026】 その際の最も確からしいサイトの決定手順は、図2に記されたようになる。ステップS1として、まず、後述の相対網羅度A<sub>15</sub>が最小のものを、利用者が検索条件として指定したカテゴリに対し、最も有効と思われるサイトとして選択する。相対網羅度A<sub>15</sub>が最小ということは、もっとも多くの該当するデータを保持しているサイトを意味する。

【0027】 次に、ステップS2では、利用者が検索条件として、指定したカテゴリに対し、以下の優先条件でソートし、複数のサイトを規定数分、選択する。

第一優先条件) 相対新規度A<sub>16</sub>が最小のもの

第二優先条件) 相対網羅度A<sub>15</sub>が最小のもの

この条件設定の理由は、可能な限り、どのサイトも扱っていないものを検索対象にするためである。

【0028】 前述の様にInitial Access Define ファイル6には、サイト1内のChild プロセス3が、常にデータ検索要求A<sub>1</sub>を転送・発行するべきアドレス情報A<sub>3</sub>が指定されているが、このInitial Access Define ファイル6は、以下の様なBNFで表される記述形式を持つ。

【0029】

【数3】

```

InitialAccessDefinefile ::= { (identifier) ' | ' } *;
(identifier) ::= (ip_string) | (dns_string) | (other);
(ip_string) ::= { (number) 3 } { ' . ' (number) 3 } 3;
(dns_string) ::= { (word) ' . ' } { (word) ' . ' } *;
(other) ::= (word) *;

```

【0030】 次に、Routing Daemon プロセス2は、サイトデータ検索要求A<sub>1</sub>を転送・発行するべきサイトのアドレス情報A<sub>3</sub>、A<sub>4</sub>にChild プロセス3を該当数分だけ立ち上げる。起動の際、各々のChild サイトプロセス3はデータ検索要求A<sub>1</sub>を転送・発行するべきアドレス情報A<sub>3</sub>、A<sub>4</sub>、並びにデータ検索要求A<sub>1</sub>の結果、転送されて来た内に保持されているデータ5'を、一時的に保持・記録するQuery Result Temp ファイル8の名称情報を引数A<sub>5</sub>として受け取る。任意のChild プロセス3がサイト1'に該データ検索要求A<sub>1</sub>と等価なデータ検索要求A<sub>8</sub>を転送・発行する際は、サイト1'内のRouting Daemon プロセス2'が受理する。その際、データ検索要求A<sub>8</sub>にはサイト1のアドレス情報が付加され、データ検索要求A<sub>8</sub>内部のHop数が1つ追加される。

【0031】 次に、Routing Daemon プロセス2は、自サイト内で管理しているデータ5からデータ検索要求A<sub>1</sub>に応じたデータのサブセットA<sub>6</sub>を受け取り、一時的に保持・記録する為、のQuery Result Temp ファイル8に該当データA<sub>7</sub>として書き込む。Query Result Temp ファイル8は、以下のBNFで表される記述形式を持ち、プロセス3ごとに1つ作成される。

【0032】

【数4】

```

QueryResultTempFile ::= <header> { <URLdescription> * } ;
<header> ::= <querytext> <amount> ;
<amount> ::= <number> 6;
<URLdescription> ::= <URL> <object_identifier> <description> ;
<URL> ::= 'http://' { <ip_string> | <dns_string> } '/' <string> ;
<object_identifier> ::= <string> ;
<description> ::= <string> ;

```

【0033】 サイト1' 内でもサイト1と同様に、Routing Daemon プロセス2' が同じ処理を行ない、サイト1' 内で管理しているデータ5' からデータ検索要求A8に応じたデータのサブセットA6を引き出し、転送データA9としてサイト内のChild プロセス3に戻す。Child プロセス3は転送データA9を受理すると、引数A5で指定されたQuery Result Temp ファイル8に該当データA10として書き込む。この処理は、起動されたChild プロセス3数分だけ実施される。

【0034】 次に、Routing Daemon プロセス2は全てのChild プロセス3が該当データA10を書き込んだことを確認すると、全てのQuery Result Temp ファイル8をマージし、そこから該当データA11を読み出す。その後、Routing Daemon プロセス2は当該データA11をデータ検索要求A1に応じた転送データA14として、データ検索要求元に転送する。以上で、一連のデータ検索要求A1は完了する。Routing Daemon プロセス2は、不要となったQuery Result Temp ファイル8を消去する際に、Cached ファイル メンテナンス9へ起動要求17を送付する。Cached ファイル メンテナンス9は起動すると、データ検索結果を格納したQuery Result Temp ファイル8から、Categorized Access Cached

ファイル7上の相対網羅度A15、相対新規度A16を更新するためのパラメータデータA12を引き出し、これを基に更新元データA13を作成し、Categorized Access Cached ファイル7上の現相対網羅度A15、現相対新規度A16を更新する。前述の様にCategorized Access Cached ファイル7とは、データ検索要求A1、A8を転送した結果、得られる一時的なデータ検索格納先であるQuery Result Temp ファイル8から、事前に定義した当該データのマイニング方式に応じて、各サイトの評価を行ない、そのアドレス情報を管理するものである。Categorized Access Cached ファイル7により、Routing Daemon プロセス2は、データ検索要求A1、A8を転送・発行する際に、検索条件として指定されるカテゴリに対し最も有効と思われるサイトのみに転送・発行先を絞り込むことが出来るので、全サイトへ単純にブロードキャスト転送を行なうよりは、遥かに効果的なルーティング転送・発行処理を実現することが出来る。Categorized Access Cached ファイル7は以下のBNFで表される記述形式を持つ。

【0035】

【数5】

```

CategorizedAccessCachedfile ::= { <registry description> ' | ' } * ;
<registry description> ::= <category identifier> <identifier> <metrix description> ;
<category identifier> ::= <string> ;
<identifier> ::= <ip_string> | <dns_string> | <other> ;
<ip_string> ::= { <number> 3 } { ' . ' <number> 3 } 3;
<dns_string> ::= { <word> ' . ' } { <word> ' . ' } * ;
<other> ::= <word> * ;
<metrix description> ::= <relative covering degree> <relative strange degree> <counter> ;
<relative covering degree> ::= <number> 6;
<relative strange degree> ::= <number> 6;
<counter> ::= <number> 6;

```

【0036】 Categorized Access Cached ファイル7上で管理される相対網羅度A15、相対新規度A16は、以下に記されるものである。

【0037】 1) 相対網羅度A15

Cached ファイル メンテナンス9は当該データのマイニング方式としてサイト1' を始めとする関連する複数のサイトがデータ検索要求A1、A8に相当する任意データ検索要求を受理した結果として戻す転送データA9、A14に対し、各サイトがどの程度、検索条件である指定カテゴリに該当するものを含んでいるかを計算

し、更新元データA13を求める。その後、その計算値である更新元データA13で、Categorized Access Cached ファイル7上で管理されるサイト、カテゴリ毎に記録している相対網羅度A15の値を更新する。相対網羅度A15の評価に関しては後述する。

【0038】 2) 相対新規度A16

Cached ファイル メンテナンス9は、前述の当該データのマイニング方式として、サイト1' を始めとする関連する複数のサイトがデータ検索要求A1、A8に相当する任意データ検索要求を受理した結果として戻す転送



データ A 9。A 14 に対し、各サイトがどの程度、検索条件である指定カテゴリに該当するもので独自かつ固有なものを含んでいるかを計算し、更新元データ A 13 を求める。その後その計算値である更新元データ A 13 で、Categorized Access Cached ファイル 7 上で管理される毎に記録している相対新規度 A 16 の値を更新する。相対新規度 A 16 の評価式に関しては後述する。

【0039】次に、当該データのマイニング方式について説明する。図 3 は、該当データのマイニング方式のフローチャートである。ステップ S 1 では、初期処理として、計算上、扱うべきサイト数を制限する。これは転送データ A 9 を戻すサイトは基本的に不特定多数であることから配慮されている。これを「サイト上限値ステップ」と呼ぶ。

【0040】ステップ S 2 は Child プロセス 3 毎に作成される Query Result Temp ファイル 8 から、〈header〉記述を集める処理である。〈header〉記述内部には、データ件数を意味する〈amount〉が記されている。

【0041】ステップ S 3 では、〈amount〉記述中の値で大きい順にソートして〈header〉記述を並べ替え、ステップ S 1 で制限されたサイト数に相当する順位のもの迄の複数サイトを処理の対象として選択する。これを「該当選択ステップ」と呼ぶ。

【0042】ステップ S 4 では、選択した複数サイトに対応する Query Result Temp ファイル 8 中の〈URLdescription〉記述を全て取り出し、1 つの〈URLdescription〉記述を 1 レコードと見なして、単純に連結する。なお、〈URLdescription〉記述の内部には、検索結果の対象を意味する〈object identifier〉記述が記されており、各々が 1 つの検索対象と見なされる。

【0043】ステップ S 5 では、〈object identifier〉記述でソートし、同じ〈object identifier〉記述を持つ異なるサイトの数を求める。以上を異なる〈object identifier〉記述毎全てにわたり実施する（ステップ S 6）。これを「該当 object 選択ステップ」と呼ぶ。

【0044】ステップ S 7 では、ステップ S 3 で選択した全サイトに対して（ステップ S 16）、各々、サイト数を変数 X とし、自身を含めた X 台のサイトに含まれる〈object identifier〉記述の度数 m (X) に関するヒストグラムを作成する。このステップ S 6 では、ステップ S 3 で選択した全サイトにわたり実施する。これを「ヒストグラム作成ステップ」と呼ぶ。

【0045】ステップ S 8 では、S 7 で求めたヒストグラムを基に網羅度及び新規度を以下の様に計算する。これを「網羅度・新規度算出ステップ」と呼ぶ。ステップ S 8 も、ステップ 7 と同様に、ステップ S 3 で選択した全サイトに対して行なう（ステップ S 17）。

【0046】

【数 6】

$$\text{網羅度} = \sum x (m(X) * X)$$

$$\text{新規度} = \sum x \{m(X)/X\}$$

【0047】ステップ S 9 では、ステップ S 8 で求めた網羅度の小さい順に選択したサイトをソートして、並べ替える。その後、小さい順に相対網羅度を 1 位、2 位、3 位と付与する。これを「相対網羅度付与ステップ」と呼ぶ。

【0048】ステップ S 10 では、ステップ S 8 で求めた新規度の小さい順に選択したサイトをソートして、並べ替える。その後、小さい順に相対新規度を 1 位、2 位、3 位と付与する。これを「相対新規度付与ステップ」と呼ぶ。

【0049】ステップ S 11 では、ステップ S 9 並びにステップ S 10 で求めた相対網羅度 A 15、相対新規度 A 16 を更新値 A 13 とする。その後、Categorized Access Cached ファイル 7 上でサイト、カテゴリ毎に管理されている現在の相対網羅度 A 15、現在の相対新規度 A 16 を読み出す。これを「既存データ読みだしステップ」と呼ぶ。

【0050】ステップ S 13 では、以下の処理を行なう。ステップ S 11 で現在の相対網羅度 A 15、現在の相対新規度 A 16 が読み出せる場合、該当する各々と更新値 A 13 との相加平均を取り、新たな相対網羅度 A 15、相対新規度 A 16 を求める。その際、相加平均の対象数を表す〈Counter〉記述値に 1 を加える。その後、Categorized Access Cached ファイル 7 に新たに算出した相対網羅度 A 15、並びに新たに算出した相対新規度 A 16 並びに〈Counter〉記述を書き込み、更新する。

【0051】ステップ S 11 で現在の相対網羅度 A 15、現在の相対新規度 A 16 が読み出せない場合は、ステップ S 9 で求めた相対網羅度 A 15、ステップ S 9 で求めた相対新規度 A 16、並びに該〈Counter〉記述を 1 として、Categorized Access Cached ファイル 7 に書き込み、更新する（ステップ S 14）。これを「更新ステップ」と呼ぶ。

【0052】ステップ S 11 並びに S 13、S 14 は、ステップ S 3 で選択した全サイト数回、処理を行う（ステップ S 15）。その後、規定回数分の処理を終えたならば、該当データのマイニング方式全体の処理を終了する。

【0053】

【発明の効果】本発明によれば、従来のように、語彙検索のみを対象とはしていないため、数値表現を含んだデータの検索が可能であり、一般的なレガシーデータをもその対象とすることが可能であるという第 1 の効果を有する。

【0054】また、本発明では、データのマイニング方式を採用しているので、動的にルーティングを実施出来、その結果、新たなサイトが登録され、それが該当デ



ータのマイニング方式の評価に対して妥当なデータを戻す場合は、新たなルーティング対象として、これを取り込むことが可能であるという第2の効果を有する。

【0055】さらに、本発明は、データ検索要求の起動状況を管理・記録するInvoke テーブルが実装されているので、検索が2重になることはないという第3の効果も有する。

【図面の簡単な説明】

【図1】 本発明のデータ検索装置の一実施例の構成図。

【図2】 本発明におけるルーティング先を決定する際の手順を示すフローチャート。

【図3】 本発明における該当データのマイニング方式の手順を示すフローチャート。

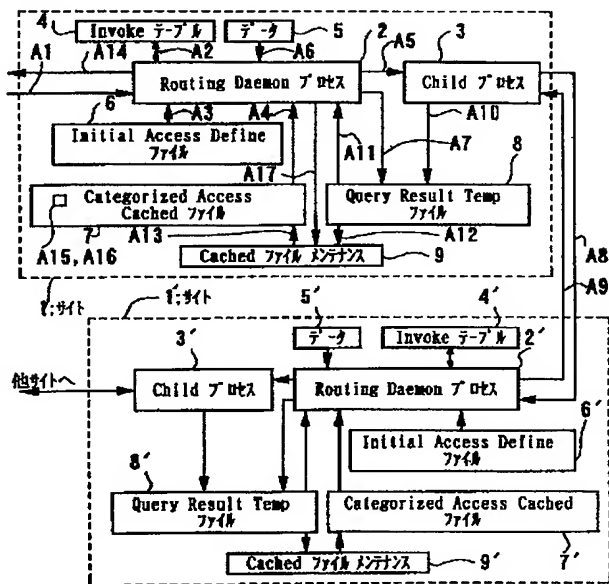
【図4】 従来方式の概要図。

【符号の説明】

1 データ転送装置（サイト）

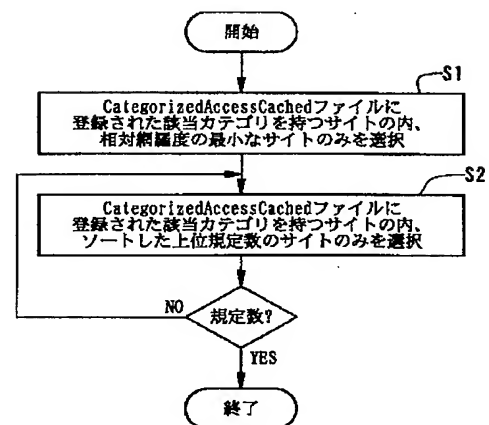
- 2 Routing Daemon プロセス
- 3 Child プロセス
- 4 Invoke テーブル
- 5 データ
- 6 Initial Access Define ファイル
- 7 Categorized Access Cached ファイル
- 8 Query Result Temp ファイル
- 9 Cached ファイル メンテナンス
- 11 Httpd プロセス
- 12 コンテントルータ
- 13 ルーティングデータベース
- 14 Refinement データベース
- 15 コンテンツラベル
- 16 サーチモジュール
- 17 WAIS プロセス
- 18 WAIS データベース

【図1】

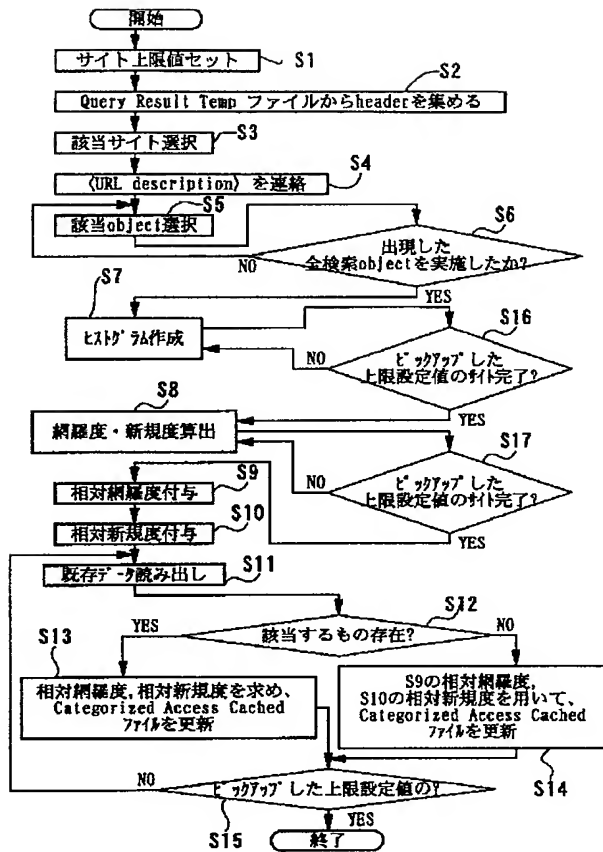


- |             |             |
|-------------|-------------|
| A1: データ検索要求 | A11: 該当データ  |
| A2: 情報      | A12: パラメータ  |
| A3: アドレス情報  | A13: 更新元データ |
| A4: アドレス情報  | A14: 転送データ  |
| A5: 引数      | A15: 相対網経度値 |
| A6: サブセット   | A16: 相対新規度値 |
| A7: 該当データ   | A17: 起動要求   |
| A8: データ検索要求 |             |
| A9: 転送データ   |             |
| A10: 該当データ  |             |

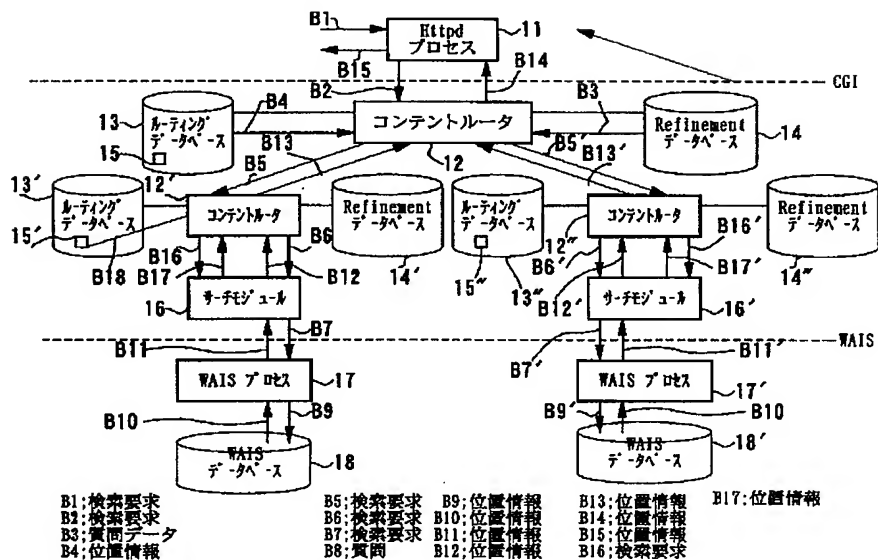
【図2】



【図 3】



【図 4】



# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-160024

(43)Date of publication of application : 12.06.2001

(51)Int.Cl.

G06F 13/00

(21)Application number : 11-342927

(71)Applicant : NEC CORP

(22)Date of filing : 02.12.1999

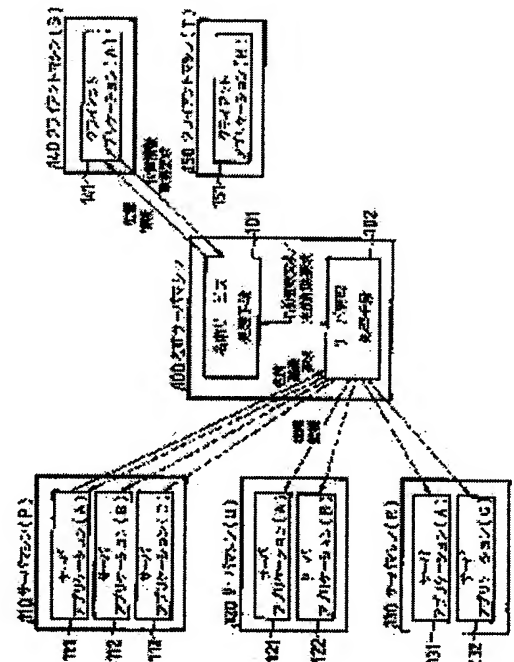
(72)Inventor : HIRAMOTO KAZUYOSHI

## (54) SYSTEM FOR SELECTING MANAGEMENT OF SERVER APPLICATION

(57)Abstract:

PROBLEM TO BE SOLVED: To allow a client application to correctly select and connect only connectable server applications.

SOLUTION: A name server processing means 101 returns position information in name service registration information, in response to a request for obtaining position information from a client application, registers name service registration information at accepting of a request for name registration, and deletes the name service registration information at accepting of a request for name deletion. A server management processing means 102 registers server management information at accepting of a request for name registration from the server application and transfers the request for name registration to the name server processing means 101, and normally monitors the state of the registered server application, and issues a request for name deletion of the server application to a server processing means 101 at the time of detecting the failure of the server application.



(4)

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号  
特開2001-160024  
(P2001-160024A)

(43)公開日 平成13年6月12日(2001.6.12)

(51)Int.Cl.<sup>7</sup>

G 0 6 F 13/00

識別記号

3 5 7

F I

G 0 6 F 13/00

テーマコード\* (参考)

3 5 7 Z 5 B 0 8 9

審査請求 有 請求項の数 8 O L (全 11 頁)

(21)出願番号 特願平11-342927

(22)出願日 平成11年12月2日(1999.12.2)

(71)出願人 000004237

日本電気株式会社

東京都港区芝五丁目7番1号

(72)発明者 平本 和好

東京都港区芝五丁目7番1号 日本電気株式会社内

(74)代理人 100088890

弁理士 河原 純一

Fターム(参考) 5B089 GA11 GA21 GB02 JA11 JA35

JB17 KA12 KB04 KB06 KC28

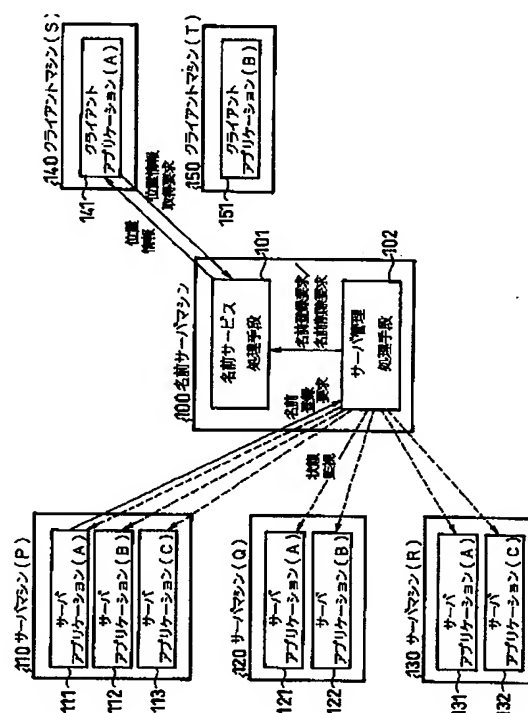
KC51 KH03 ME06

(54)【発明の名称】 サーバアプリケーションの管理選択方式

(57)【要約】

【課題】接続可能なサーバアプリケーションのみをクライアントアプリケーションが正しく選択して接続できるようにする。

【解決手段】名前サービス処理手段101は、クライアントアプリケーションからの位置情報取得要求に対して登録している名前サービス登録情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス登録情報を登録し、名前削除要求を受け付けて名前サービス登録情報を削除する。サーバ管理処理手段102は、サーバアプリケーションからの名前登録要求を受け付けてサーバ管理情報を登録するとともに名前サービス処理手段101に名前登録要求を転送し、登録されたサーバアプリケーションに対しては定期的に状態監視を行い、サーバアプリケーションの障害を検出した場合に名前サービス処理手段101に対して当該サーバアプリケーションの名前削除要求を発行する。



**【特許請求の範囲】**

【請求項 1】サーバアプリケーションを提供する任意の数のサーバマシン、サーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン、およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおいて、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段と、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段とを備えることを特徴とするサーバアプリケーションの管理選択方式。

【請求項 2】前記サーバ管理処理手段が、サーバアプリケーションからの名前登録要求を受け付けて監視対象情報にリンクして名前サービス登録情報を登録し該名前登録要求を前記名前サービス処理手段に転送する名前登録部と、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録部を介して前記名前サービス処理手段に発行するサーバ監視部とを含んで構成される請求項 1 記載のサーバアプリケーションの管理選択方式。

【請求項 3】前記名前サービス登録情報が、サーバアプリケーションの名前と、該サーバアプリケーションの位置情報とから構成される請求項 1 または 2 記載のサーバアプリケーションの管理選択方式。

【請求項 4】前記名前サービス管理情報が、サーバアプリケーションの名前と、該名前前にリンクされたサーバアプリケーションの位置情報とから構成される請求項 1 または 2 記載のサーバアプリケーションの管理選択方式。

【請求項 5】前記監視対象情報が、監視対象の単位をサーバアプリケーションとした場合にサーバマシンの名前とサーバアプリケーションの名前との組である請求項 1 または 2 記載のサーバアプリケーションの管理選択方式。

【請求項 6】前記監視対象情報が、監視対象の単位をサーバマシンとした場合にサーバマシンの名前である請求項 1 または 2 記載のサーバアプリケーションの管理選択方式。

【請求項 7】コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求

を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段、およびサーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段として機能させるためのプログラムを記録した記録媒体。

【請求項 8】コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録手段、および各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録手段を介して前記名前サービス処理手段に発行するサーバ監視手段として機能させるためのプログラムを記録した記録媒体。

**【発明の詳細な説明】****【0001】**

【発明の属する技術分野】本発明はサーバアプリケーションの管理選択方式に関し、特にサーバアプリケーションを搭載する任意の数のサーバマシン、サーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン、およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおけるサーバアプリケーションの管理選択方式に関する。

**【0002】**

【従来の技術】名前サーバマシンを利用するネットワークシステムでは、サーバアプリケーションが名前および位置情報を名前サーバマシンに登録し、クライアントアプリケーションは名前をキーに名前サーバマシンにサーバアプリケーションの位置情報を取得要求し、返却されたサーバアプリケーションの位置情報を用いてサーバアプリケーションに接続して運用を行う。

【0003】従来の名前サーバマシンを利用するネットワークシステムでは、サーバアプリケーションが名前および位置情報の登録後に障害等を起こし接続不可となった場合でも、名前サーバマシンにサーバアプリケーションの名前および位置情報が残り、クライアントアプリケーションからのサーバアプリケーションの位置情報取得要求時に接続不可のサーバアプリケーションの位置情報

を返却してしまうという不具合があった。

【0004】そこで、特開平6-68007号公報に開示された「クライアント・サーバ連結装置」では、サーバアプリケーションが位置情報を名前サーバマシンに定期的に再登録することで、名前サーバマシンにおけるサーバアプリケーションの位置情報を最適化している。また、サーバアプリケーションの障害等をクライアントアプリケーションで認識した場合は、クライアントアプリケーションがサーバアプリケーションの位置情報を名前サーバマシンから削除することによっても名前サーバマシンにおけるサーバアプリケーションの位置情報を更新している。

【0005】

【発明が解決しようとする課題】しかし、上述した従来の技術には、次のような問題点があった。

【0006】第1の問題点は、障害等を起こして接続不可になったサーバアプリケーションへの接続を回避するためには、名前サーバマシンを始めとする各処理手段に機能を追加する等、システム全体に変更を加えなくては実現できないことである。その理由は、システムの各部で問題回避の処理を行っているためである。

【0007】第2の問題点は、状態維持のためのコストがかかることである。その理由は、すべてのサーバアプリケーションが本来一度で済むべき名前登録処理を定期的に繰り返すため、それによるシステム負荷およびネットワーク負荷がサーバアプリケーション数および接続時間に比例してかかってしまうためである。

【0008】第3の問題点は、サーバアプリケーションの名前が不当に削除される可能性があることである。その理由は、クライアントアプリケーションでサーバアプリケーションの不正状態を検出した場合に、そのままサーバアプリケーションの名前を削除してしまうからである。このため、クライアントアプリケーションの状態の如何によっては、正常に動作しているサーバアプリケーションの名前をも削除しかねなかった。

【0009】本発明の目的は、接続可能なサーバアプリケーションのみをクライアントアプリケーションが正しく選択して接続できるようにしたサーバアプリケーションの管理選択方式を提供することにある。

【0010】

【課題を解決するための手段】本発明のサーバアプリケーションの管理選択方式は、サーバアプリケーションを提供する任意の数のサーバマシン、これらサーバアプリケーションに接続して運用するクライアントアプリケーションを搭載する任意の数のクライアントマシン、およびサーバアプリケーションの位置情報を管理する名前サービスを提供する名前サーバマシンを含んで構成されるネットワークシステムにおいて、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要

求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段と、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段とを備えることを特徴とする。

【0011】さらに、本発明のサーバアプリケーションの管理選択方式は、前記サーバ管理処理手段が、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録部と、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録部を介して前記名前サービス処理手段に発行するサーバ監視部とを含んで構成されることを特徴とする。

【0012】一方、本発明の記録媒体は、コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段、およびサーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送するとともに、各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前サービス処理手段に発行するサーバ管理処理手段として機能させるためのプログラムを記録する。

【0013】また、本発明の記録媒体は、コンピュータを、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の位置情報を返却するとともに、名前登録要求を受け付けて名前サービス管理情報を登録し、名前削除要求を受け付けて名前サービス管理情報を削除する名前サービス処理手段、サーバアプリケーションからの名前登録要求を受け付けて監視対象毎に名前サービス登録情報を登録して該名前登録要求を前記名前サービス処理手段に転送する名前登録手段、および各監視対象を定期的に状態監視し、監視対象の障害を検出した場合に該監視対象のサーバアプリケーションの名前削除要求を前記名前登録手段を介して前記名前サービス処理手段に発行するサーバ監視手段として機能させるためのプログラムを記録する。

【0014】本発明のサーバアプリケーションの管理選択方式では、名前サーバマシンは、通常の名前サービス

を行う名前サービス処理手段の他に、サーバ管理処理手段を有し、名前サービス処理手段に登録された各サーバマシン上の各サーバアプリケーションは、サーバ管理処理手段によって定常的に状態監視がなされる。これにより、あるサーバアプリケーションが障害等により接続不可となった場合には、サーバ管理処理手段が当該サーバアプリケーションの名前削除要求を名前サービス処理手段に速やかに発行するため、クライアントアプリケーションからのサーバアプリケーションの位置情報取得要求に対して、名前サーバマシンが接続不可となったサーバアプリケーションの位置情報をクライアントアプリケーションに返却することがなくなる。クライアントアプリケーションにおいては、名前サービス処理手段からサーバアプリケーションの位置情報取得要求は、従来の名前サーバマシンへの位置情報取得要求と同様に発行することが可能である。このようにして、従来の名前サービスを利用したネットワークシステムの形態に極力手を加えずに、接続可能なサーバアプリケーションの位置情報のみを名前サービス処理手段に残すことを可能にする。また、監視対象を調整することにより、監視のための通信負荷を低減したりシステム全体の性能にも配慮することが可能である。

#### 【0015】

【発明の実施の形態】以下、本発明の実施の形態について図面を参照して詳細に説明する。

【0016】図1は、本発明の第1の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムの要部を示すブロック図である。このネットワークシステムは、サーバアプリケーション

(A) 111、サーバアプリケーション (B) 112 およびサーバアプリケーション (C) 113 を搭載するサーバマシン (P) 110 と、サーバアプリケーション

(A) 121 およびサーバアプリケーション (B) 122 を搭載するサーバマシン (Q) 120 と、サーバアプリケーション (A) 131 およびサーバアプリケーション (C) 132 を搭載するサーバマシン (R) 130 と、サーバアプリケーション (A) 111、121 および131に接続して運用するクライアントアプリケーション (A) 141 を搭載するクライアントマシン (S) 140 と、サーバアプリケーション (B) 112 および122に接続して運用するクライアントアプリケーション (B) 151 を搭載するクライアントマシン (T) 150 と、各サーバアプリケーションの名前サービス管理情報を登録し、クライアントアプリケーション (A) 141 およびクライアントアプリケーション (B) 151 からの位置情報取得要求に応じて接続可能な1つのサーバアプリケーションの位置情報を返却する名前サービスを提供する名前サーバマシン100とから、その主要部が構成されている。なお、括弧書きのアルファベットは名前を表し、同じ名前のサーバアプリケーションは同一

のアプリケーションであり、いずれも同じ名前のクライアントアプリケーションから接続されて運用可能なものである。

【0017】名前サーバマシン100は、名前サービス処理手段101と、サーバ管理処理手段102とを含んで構成されている。

【0018】名前サービス処理手段101は、サーバ管理処理手段102からの名前登録要求を受け付け、名前サービス管理情報を登録する。また、名前サービス処理手段101は、サーバ管理処理手段102からの名前削除要求を受け付け、名前サービス管理情報を削除する。さらに、名前サービス処理手段101は、クライアントアプリケーションからの位置情報取得要求に対して名前サービス管理情報中の名前にリンクされた位置情報のうちの1つを返却する。

【0019】サーバ管理処理手段102は、各サーバアプリケーションからの名前登録要求を受け付け、監視対象情報にリンクさせてサーバアプリケーションの名前および位置情報からなる名前サービス登録情報を登録するとともに、名前登録要求を名前サービス処理手段101に転送する。また、サーバ管理処理手段102は、監視対象情報に登録された監視対象（サーバマシン、サーバアプリケーション等）に対して定常的に状態監視を行う。さらに、サーバ管理処理手段102は、状態監視により監視対象の障害等を検出した場合、当該監視対象のサーバアプリケーションの名前削除要求を名前サービス処理手段101に速やかに発行する。

【0020】図2を参照すると、サーバ管理処理手段102は、各サーバアプリケーションより名前登録要求を受け付けて名前登録要求を名前サービス処理手段101に転送する名前登録部1021と、監視対象情報を基に各監視対象を定期的に状態監視するサーバ監視部1022と、サーバ管理情報を格納するサーバ管理情報格納部1023とを備えている。

【0021】名前登録部1021は、あらかじめ監視対象情報310（図3参照）をサーバ管理情報格納部1023に登録する。また、名前登録部1021は、サーバアプリケーションからの名前登録要求を受け付けると

（図2①参照）、名前登録要求から名前サービス登録情報320（図3参照）を作成し監視対象情報310にリンクさせてサーバ管理情報格納部1023に登録する

（図2②参照）。すなわち、名前登録部1021は、名前サービス登録情報320を、予め規定しておいた監視対象の単位（サーバマシン、サーバアプリケーション等）に従って監視対象情報310にリンクさせてサーバ管理情報としてサーバ管理情報格納部1023に登録する。この後、名前登録部1021は、サーバアプリケーションからの名前登録要求を名前サービス処理手段101に転送する（図2③参照）。

【0022】サーバ監視部1022は、サーバ管理情報

格納部 1023 のサーバ管理情報を検索して監視対象情報 310 を取得し (図 2④参照)、監視対象情報 310 に基づいて各監視対象を状態監視する (図 2⑤参照)。状態監視時に監視対象の障害等を検出した場合は、サーバ監視部 1022 は、サーバアプリケーションに代わって名前削除要求を名前登録部 1021 を介して名前サービス処理手段 101 に発行する (図 2⑥参照)。

【0023】図 3 を参照すると、サーバ管理情報格納部 1023 のサーバ管理情報は、監視対象情報 310 と、監視対象情報 310 にリンクされた 1 つ以上の名前サービス登録情報 320 とから構成されている。

【0024】監視対象情報 310 は、サーバ管理処理手段 102 が監視対象を状態監視するために必要な情報を格納する。格納される情報は、監視対象の種類によって変化する。監視対象の単位を最小単位であるサーバアプリケーションとした場合は、サーバアプリケーションの固有情報となるサーバマシンの名前とサーバアプリケーションの名前との組が監視対象情報となる。また、監視対象の単位をサーバマシンとした場合は、サーバマシンの名前が監視対象情報となる。その他、サーバマシングループ、サーバアプリケーショングループ等を監視対象の単位とすることが可能である。

【0025】名前サービス登録情報 320 は、名前登録要求があったサーバアプリケーションの名前と、その位置情報との組で構成される。なお、位置情報の後の括弧内のアルファベットは、サーバマシンの名前およびサーバアプリケーションの名前の組からなる位置情報の値を示す (以下同様)。名前サービス登録情報 320 は、新規にサーバアプリケーションから名前登録要求を受け付けた場合、必ず作成される情報であり、サーバ管理処理手段 102 が各サーバアプリケーションより受け付けた名前登録要求の内容の記録であるとともに、名前サービス処理手段 101 に登録した内容の記録である。

【0026】図 2 を参照すると、名前サービス処理手段 101 は、名前サービス管理情報格納部 1011 を備えている。

【0027】図 4 を参照すると、名前サービス管理情報格納部 1011 は、サーバ管理情報格納部 1023 に登録されたサーバアプリケーションの名前 410 と、その名前 410 を持つ 1 つ以上のサーバアプリケーションの位置情報 420 とを管理する。1 つの名前 410 に対して位置情報 420 を複数持つ場合、クライアントアプリケーションから位置情報取得要求された名前 410 に対して管理している位置情報 420 のうちの 1 つを巡回方式で選択して返却する。

【0028】図 5 を参照すると、名前登録部 1021 の名前登録時の処理は、サーバ管理情報更新ステップ 21 と、名前登録要求転送ステップ 22 と、名前登録要求受け付け判定ステップ 23 と、状態監視開始ステップ 24 と、サーバ管理情報巻き戻しステップ 25 とからなる。

【0029】図 6 を参照すると、サーバ監視部 1022 の障害検出時の処理は、接続不可監視対象検出ステップ 31 と、名前サービス登録情報検索ステップ 32 と、名前削除要求発行ステップ 33 と、監視対象除外ステップ 34 とからなる。

【0030】次に、このように構成された第 1 の実施の形態に係るサーバアプリケーションの管理選択方式の動作について詳細に説明する。

【0031】まず、名前登録時の処理について説明する。

【0032】サーバ管理処理手段 102 の名前登録部 1021 は、各サーバアプリケーションからの名前登録要求を待ち受けている。

【0033】名前登録部 102 は、あるサーバアプリケーションから名前登録要求を受け付けると (図 2①参照)、名前登録要求中のサーバアプリケーションの名前および位置情報を名前サービス登録情報 320 として該当する監視対象情報 310 にリンクさせてサーバ管理情報格納部 1023 に登録する (図 2②参照) (ステップ 21)。

【0034】次に、名前登録部 102 は、受け付けた名前登録要求を名前サービス処理手段 101 に転送する (図 2③参照) (ステップ 22)。

【0035】名前サービス処理手段 101 は、名前登録要求を受け付けると、名前登録要求中のサーバアプリケーションの名前が名前サービス管理情報格納部 1011 にすでに登録されていれば、名前登録要求中のサーバアプリケーションの位置情報を該当する名前にリンクさせて名前サービス管理情報格納部 1011 に登録する。名前登録要求中のサーバアプリケーションの名前が名前サービス管理情報格納部 1011 に登録されていなければ、名前サービス処理手段 101 は、名前登録要求中のサーバアプリケーションの名前および位置情報をリンクさせて名前サービス管理情報として名前サービス管理情報格納部 1011 に登録する。この後、名前サービス処理手段 101 は、名前登録要求に基づく名前登録が正常に行われたか否かを名前登録結果として名前登録部 1021 に返却する。

【0036】名前登録部 1021 は、名前サービス処理手段 101 から返却された名前登録結果に基づいて名前登録要求が受け付けられたかどうかを判断し (ステップ 23)、受け付けられたのであれば、要求元のサーバアプリケーションに名前の登録完了を返却し、以後、監視対象の状態監視を開始する (ステップ 24)。

【0037】一方、名前登録要求が受け付けられなかったのであれば、名前登録部 1021 は、要求元のサーバアプリケーションに名前の登録失敗を通知し、サーバ管理情報格納部 1023 のサーバ管理情報を巻き戻す (ステップ 25)。

【0038】次に、障害等による接続不可検出時の処理



について説明する。

【0039】サーバ管理処理手段102のサーバ監視部1022は、サーバ管理情報格納部1023のサーバ管理情報を検索して監視対象情報310を取得し（図2④参照）、監視対象情報310に基づいて監視対象を定期的に状態監視する（図2⑤参照）。

【0040】サーバ監視部1022は、ある監視対象が障害等により接続不可になったことを検出すると（ステップ31）、サーバ管理情報格納部1023から当該監視対象の監視対象情報310にリンクされた名前サービス登録情報320を検索し（ステップ32）、この名前サービス登録情報320中の名前の名前削除要求を名前登録部1021を介して名前サービス処理手段101に発行する（図2⑥参照）（ステップ33）。

【0041】名前サービス処理手段101は、名前削除要求を受け付けると、名前削除要求中のサーバアプリケーションの名前が名前サービス管理情報格納部1011にすでに登録されていれば、名前登録要求中のサーバアプリケーションの名前410および該名前410にリンクされた全ての位置情報420を名前サービス管理情報格納部1011から削除する。この後、名前サービス処理手段101は、名前削除要求に基づく名前削除が正常に行われたか否かを名前削除結果として名前登録部1021に返却する。

【0042】名前登録部1021は、名前サービス処理手段101から返却された名前削除結果を受け付けると、サーバ監視部1022に転送する。

【0043】サーバ監視部1022は、名前削除結果を受け付けると、サーバ管理情報格納部1023のサーバ管理情報中の該当する監視対象情報310およびそれにリンクされた名前サービス登録情報320を削除し、以降の監視対象から除外する（ステップ34）。

【0044】次に、具体例を用いてより詳細に説明する。

【0045】（1）まず、監視対象がサーバアプリケーションである場合を例にとって説明する。

【0046】図7に示すように、サーバアプリケーション(A)111、121および131がサーバマシン(P)110、サーバマシン(Q)120およびサーバマシン(R)130上でそれぞれ稼動し、サーバアプリケーション(B)112および122がサーバマシン(P)110およびサーバマシン(Q)120上でそれぞれ稼動し、サーバアプリケーション(C)113および132がサーバマシン(P)110およびサーバマシン(R)130上でそれぞれ稼動しているものとする。また、クライアントマシン(S)140上で稼動するクライアントアプリケーション(A)141は、サーバアプリケーション(A)に接続して動作するものとする。

【0047】図7の場合、監視対象の単位をサーバアプリケーションとしているが、これは、図8に示すよう

に、監視対象情報をサーバアプリケーション単位に構築することによって実現される。すなわち、監視対象情報310は、サーバマシンの名前およびサーバアプリケーションの名前の組で構成され、新規に名前登録要求があったサーバアプリケーション毎に1つ作成される。同時に登録内容である名前サービス登録情報320が1対1で対応付けられて管理される。

【0048】まず、クライアントアプリケーション(A)141が、名前サーバマシン100上の名前サービス処理手段101に対してサーバアプリケーション(A)の位置情報取得要求を発行したものとする。

【0049】名前サービス処理手段101は、通常状態では、クライアントアプリケーション(A)141からのサーバアプリケーション(A)の位置情報取得要求に対して、名前サービス管理情報格納部1011に格納されている名前410にリンクされている位置情報420のうちの1つを返却する。具体的には、図4に例示するように、名前サービス管理情報格納部1011には、サーバアプリケーション(A)の名前(A)にリンクしてサーバアプリケーション(A)111の位置情報(PA)、サーバアプリケーション(A)121の位置情報(QA)およびサーバアプリケーション(A)131の位置情報(RA)の3つの位置情報が登録されているので、名前サービス処理手段101は、3つの位置情報(PA)、(QA)および(RA)のうちから1つを選択してクライアントアプリケーション(A)141に返却する。

【0050】ここで、サーバマシン(Q)120のサーバアプリケーション(A)121が障害等により接続不可となったものとする。

【0051】各サーバアプリケーションは、サーバ管理処理手段102のサーバ監視部1022によって監視されている。

【0052】サーバ監視部1022は、サーバマシン(Q)120のサーバアプリケーション(A)121が障害等により接続不可になったことを検出すると（ステップ31）、サーバマシン(Q)120の名前(Q)およびサーバアプリケーション(A)121の名前(A)をキーとしてサーバ管理情報格納部1023に格納されているサーバ管理情報から名前サービス登録情報320を検索し（ステップ32）、検索された名前サービス登録情報320に基づいてサーバマシン(Q)120のサーバアプリケーション(A)121の名前サービス管理情報を削除する名前削除要求を作成し名前登録部1021を介して名前サービス処理手段101に発行する（ステップ33）。

【0053】名前サービス処理手段101は、名前削除要求を受け付けると、名前削除要求の対象である名前(A)および該名前(A)にリンクされた位置情報(QA)からなる名前サービス管理情報を名前サービス管理

情報格納部 1011 から削除する。すなわち、サーバマシン (Q) 120 のサーバアプリケーション (A) 121 の名前サービス管理情報を削除する。

【0054】 名前サービス処理手段 101 から名前削除結果が返却されると、サーバ監視部 1022 は、サーバ管理情報格納部 1023 からサーバマシン (Q) 120 の監視対象情報 310 にリンクされたサーバアプリケーション (A) 121 の名前 (A) および位置情報 (Q A) からなる名前サービス登録情報を削除する (ステップ 34)。

【0055】 よって、この後、名前サービス処理手段 101 は、クライアントアプリケーション (A) 141 からサーバアプリケーション (A) の位置情報取得要求があっても、サーバマシン (Q) 120 上のサーバアプリケーション (A) 121 の位置情報 (Q A) を選択して返却することではなく、サーバマシン (P) 110 上のサーバアプリケーション (A) 111 の位置情報 (P A) かサーバマシン (R) 130 上のサーバアプリケーション (A) 131 の位置情報 (R A) かを選択して返却する。

【0056】 (2) 次に、監視対象の単位がサーバマシンである場合を例にとって説明する。

【0057】 図 9 に示すように、サーバアプリケーション (A) 111, 121 および 131 がサーバマシン (P) 110, サーバマシン (Q) 120 およびサーバマシン (R) 130 上でそれぞれ稼動し、サーバアプリケーション (B) 112 および 122 がサーバマシン (P) 110 およびサーバマシン (Q) 120 上でそれぞれ稼動し、サーバアプリケーション (C) 113 および 132 がサーバマシン (P) 110 およびサーバマシン (R) 130 上でそれぞれ稼動しているものとする。

【0058】 図 9 の場合、監視対象の単位をサーバマシンとしているが、これは、図 10 に示すように、監視対象情報 310 をサーバマシンを単位に構築することによって実現する。すなわち、監視対象情報 310 は、サーバマシンの名前からなり、新規に名前登録要求があったサーバマシン毎に作成される。また、このときに、1 つ目の名前サービス登録情報 320 が監視対象情報 310 にリンクされる。以降、すでに名前登録要求の実績があるサーバマシンからの名前登録要求の受付時には、前に作成された監視対象情報 310 に新しい名前サービス登録情報 320 がリンクされる。

【0059】 いま、クライアントアプリケーション (A) 141 が、サーバアプリケーション (A) の位置情報取得要求を名前サービス処理手段 101 に発行したとする。

【0060】 名前サービス処理手段 101 は、通常状態では、クライアントアプリケーション (A) 141 からのサーバアプリケーション (A) の位置情報取得要求に対して、名前サービス管理情報格納部 1011 に名前サ

ービス管理情報が格納されているサーバマシン (P) 110 の位置情報 (P A), サーバマシン (Q) 120 の位置情報 (Q A), およびサーバマシン (R) 130 の位置情報 (R A) のうちから 1 つを選択してクライアントアプリケーション (A) 141 に返却する。

【0061】 サーバ管理処理手段 102 のサーバ監視部 1022 は、サーバマシン (P) 110, サーバマシン (Q) 120 およびサーバマシン (R) 130 を状態監視している。本例では、監視対象の単位がサーバアプリケーションでなくサーバマシンであるため、監視対象が広がることにより、状態監視に必要な通信コストが抑えられる。

【0062】 ここで、サーバマシン (Q) 120 が障害等によりダウンして接続不可になったものとする。

【0063】 サーバ監視部 1022 は、サーバマシン (Q) 120 が障害等によりダウンして接続不可となったことを検出すると (ステップ 31)、サーバ管理情報格納部 1023 のサーバ管理情報からサーバマシン (Q) 120 の監視対象情報 310 にリンクされたすべての名前サービス登録情報 320 を検索し (ステップ 32)、サーバマシン (Q) 120 のすべてのサーバアプリケーション (A) 121 およびサーバアプリケーション (B) 122 の名前サービス管理情報を削除する名前削除要求を名前登録部 1021 を介して名前サービス処理手段 101 に発行する (ステップ 33)。

【0064】 名前サービス処理手段 101 は、名前削除要求を受け付けると、名前サービス管理情報格納部 1011 の名前サービス管理情報からサーバマシン (Q) 120 のサーバアプリケーション (A) 121 およびサーバアプリケーション (B) 122 の名前サービス管理情報を削除する。

【0065】 名前サービス処理手段 101 から名前削除結果が返却されると、サーバ監視部 1022 は、サーバ管理情報格納部 1023 からサーバ管理情報を削除する (ステップ 34)。

【0066】 よって、この後、クライアントアプリケーション (A) 140 から、サーバアプリケーション (A) に対する位置情報取得要求があっても、サーバマシン (Q) 120 のサーバアプリケーション (A) 121 を選択することではなく、サーバマシン (P) 110 のサーバアプリケーション (A) 111 かサーバマシン (R) 130 のサーバアプリケーション (A) 131 かを選択して返却することになる。

【0067】 次に、本発明の第 2 の実施の形態について図面を参照して説明する。

【0068】 図 11 を参照すると、本発明の第 2 の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムは、図 1 に示した第 1 の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムに対して、名前

サーバマシン 100 が名前サーバプログラムを記録した記録媒体 200 を備える点だけが異なっている。したがって、対応する手段等には同一の符号を付してそれらの詳しい説明を省略する。なお、記録媒体 200 は、磁気ディスク、半導体メモリ、その他の記録媒体であってよい。

【0069】このように構成された第 2 の実施の形態に係るサーバアプリケーションの管理選択方式では、名前サーバプログラムが記録媒体 200 からコンピュータでなる名前サーバマシン 100 に読み込まれ、名前サービス処理手段 101 およびサーバ管理処理手段 102 として名前サーバマシン 100 の動作を制御する。名前サーバマシン 100 の詳しい動作は、第 1 の実施の形態に係るサーバアプリケーションの管理選択方式における場合と全く同様になるので、その詳しい説明を割愛する。

【0070】ところで、上記各実施の形態では、監視対象の単位をサーバアプリケーションまたはサーバマシンとした場合について説明したが、監視対象の単位をサーバアプリケーショングループとしても、サーバマシングループとしても、本発明を同様に適用することができる。例えば、監視対象の単位をサーバアプリケーショングループとした場合、他のサーバアプリケーションに大きな影響を与えるサーバアプリケーションの障害を検出したときに、これに依存する全てのサーバアプリケーションの名前を削除することが可能となる。また、監視対象の単位をサーバマシングループとした場合、サーバマシンの障害を検出した際に、当該サーバマシングループから登録された全てのサーバアプリケーションの名前を削除することも可能である。このように、監視対象の単位を調整したり、それらを組み合わせて使うことで、より柔軟で効率的な監視処理を行うことが可能となる。

#### 【0071】

【発明の効果】第 1 の効果は、障害等により接続不可となったサーバアプリケーションにクライアントアプリケーションが接続にいくことを未然に回避できることである。その理由は、サーバ管理処理手段により接続不可のサーバアプリケーションの名前は名前サーバマシンから削除されており、クライアントアプリケーションは接続不可のサーバアプリケーションの位置情報を取得することがないためである。

【0072】第 2 の効果は、既存の名前サービスを利用するネットワークシステムにそのまま適用できることにある。その理由は、クライアントアプリケーションに特別な機能を追加することも、クライアントマシンに特別な処理手段を追加することもないため、既存のクライアント環境に手を加えることなくそのまま使い続けることが可能である。また、名前サービスも登録要求元が各サーバアプリケーションからサーバ管理処理手段に変わるだけなので、既存の名前サービスにも影響を与えない。サーバアプリケーションにおいても、位置情報の

登録処理に関しては、登録要求元が名前サービス処理手段からサーバ管理処理手段にかわる以外に影響はない。

【0073】第 3 の効果は、監視対象を切り替えることで効率の良い監視ができることである。その理由は、監視対象をサーバアプリケーション毎、サーバアプリケーショングループ毎、サーバマシン毎、サーバマシングループ毎というように切り替えることにより、1 つ障害で効率よく名前サービスへの更新を行うことができるためである。

#### 【図面の簡単な説明】

【図 1】本発明の第 1 の実施の形態に係るサーバアプリケーションの管理選択方式が適用されたネットワークシステムの要部を示すブロック図である。

【図 2】図 1 中の名前サーバマシンのより詳細な構成を示すブロック図である。

【図 3】図 2 中のサーバ管理情報格納部の内容を例示する図である。

【図 4】図 2 中の名前サービス管理情報格納部の内容を例示する図である。

【図 5】図 2 中の名前登録部の名前登録時の処理を示すフローチャートである。

【図 6】図 2 中のサーバ監視部の障害検出時の処理を示すフローチャートである。

【図 7】図 1 に示した第 1 の実施の形態に係るサーバアプリケーションの管理選択方式の運用例を示す図である。

【図 8】図 7 に示した運用例でのサーバ管理情報格納部の内容例を示す図である。

【図 9】図 1 に示した第 1 の実施の形態に係るサーバアプリケーションの管理選択方式の他の運用例を示す図である。

【図 10】図 9 に示した運用例でのサーバ管理情報格納部の内容例を示す図である。

【図 11】本発明の第 2 の実施の形態に係るサーバアプリケーションの管理選択方式の構成を示すブロック図である。

#### 【符号の説明】

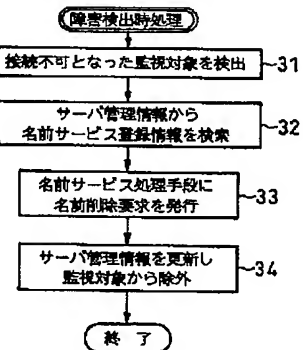
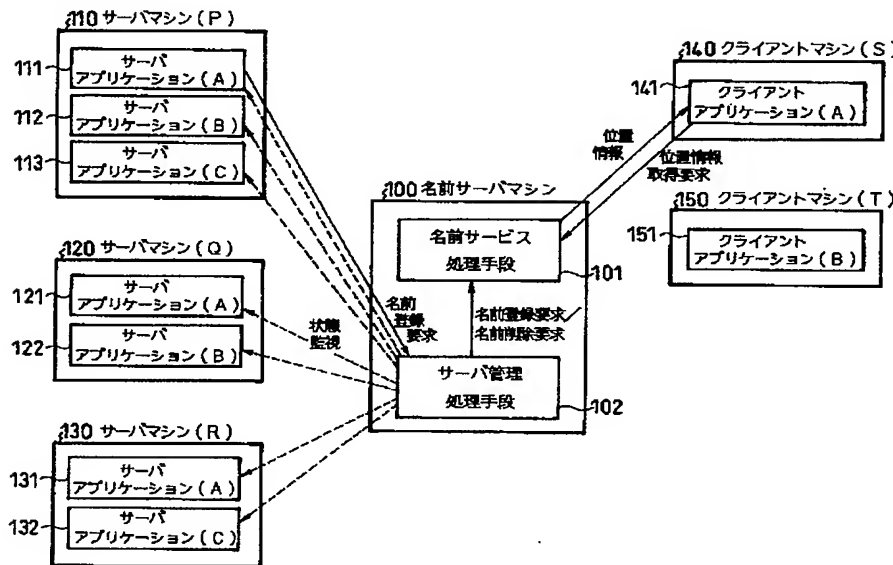
- 100 名前サーバマシン
- 101 名前サービス処理手段
- 102 サーバ管理処理手段
- 110 サーバマシン (P)
- 111, 121, 131 サーバアプリケーション (A)
- 112, 122 サーバアプリケーション (B)
- 113, 132 サーバアプリケーション (C)
- 120 サーバマシン (Q)
- 130 サーバマシン (R)
- 140 クライアントマシン (S)
- 141 クライアントアプリケーション (A)
- 150 クライアントマシン (T)

151 クライアントアプリケーション (B)  
 310 監視対象情報  
 320 名前サービス登録情報  
 410 名前  
 420 位置情報

1011 名前サービス管理情報格納部  
 1021 名前登録部  
 1022 サーバ監視部  
 1023 サーバ管理情報格納部

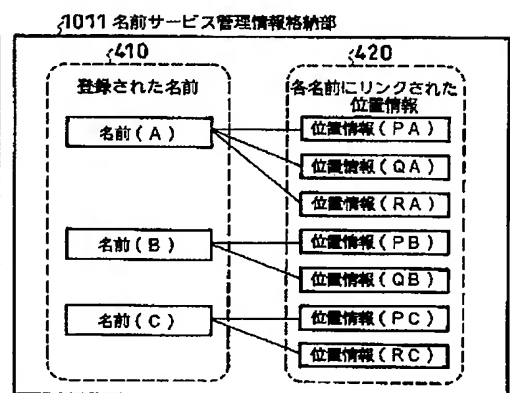
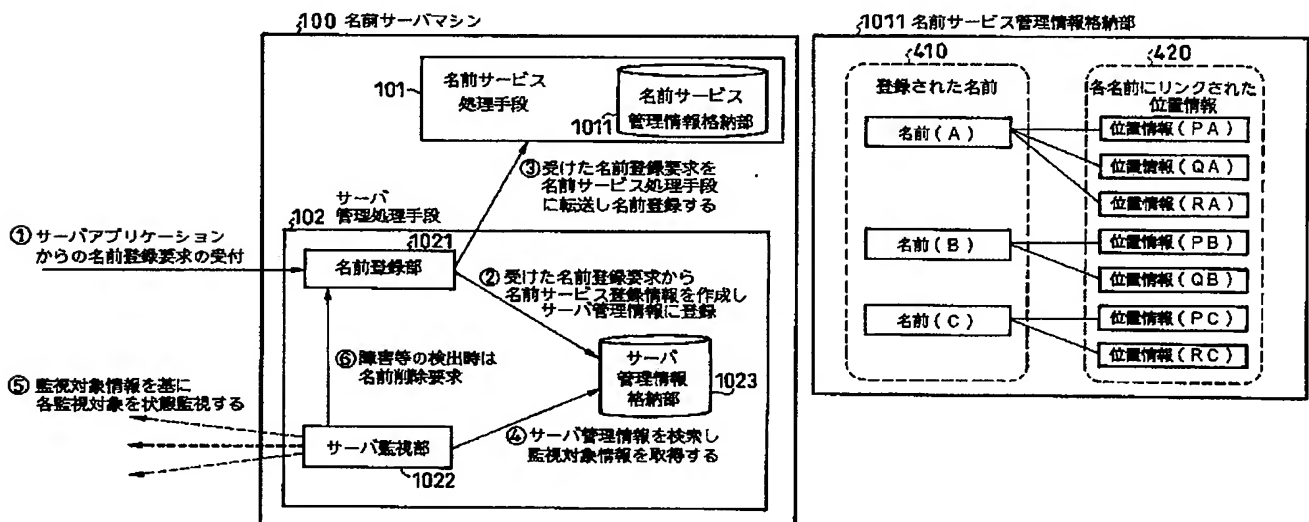
【図 1】

【図 6】

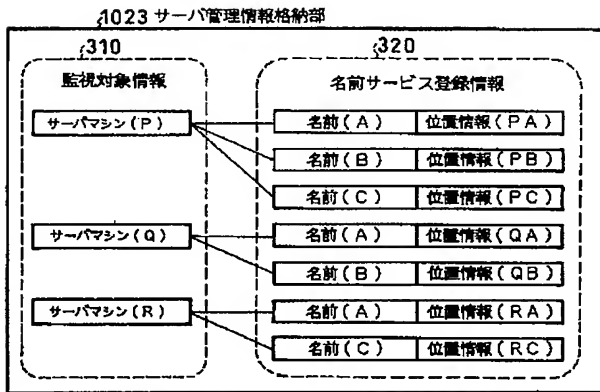


【図 2】

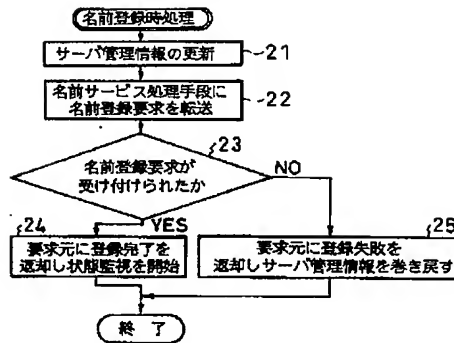
【図 4】



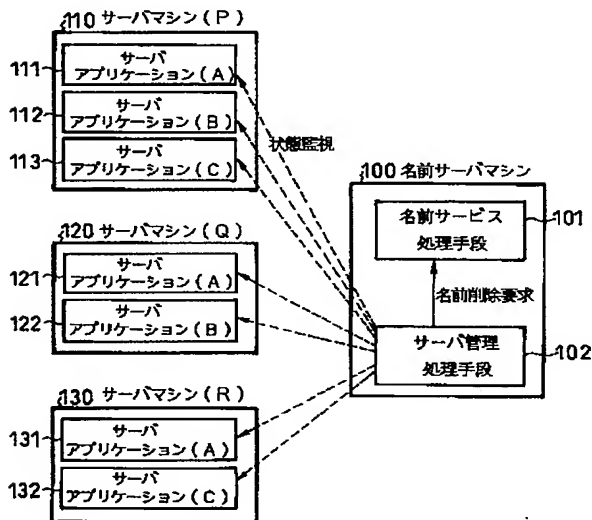
【図 3】



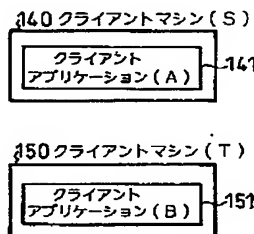
【図 5】



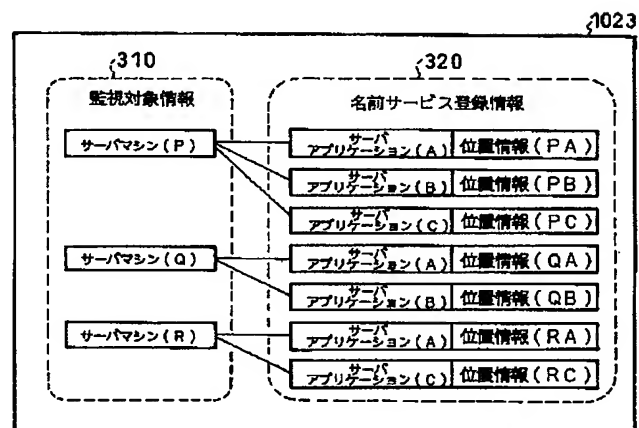
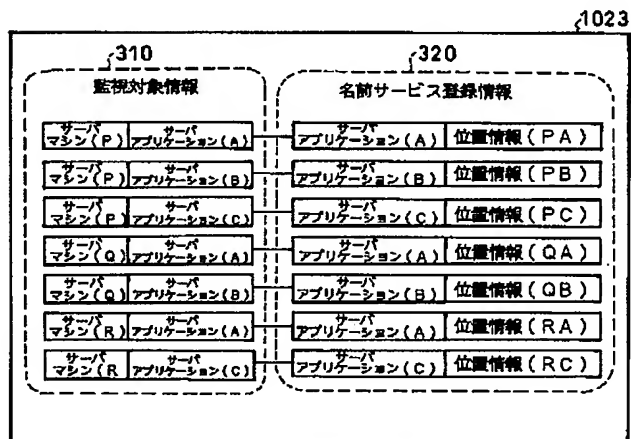
【図 7】



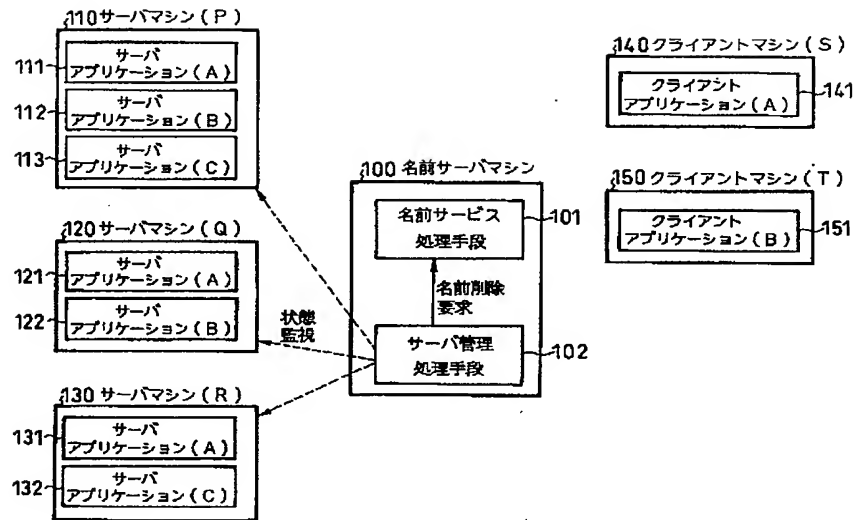
【図 8】



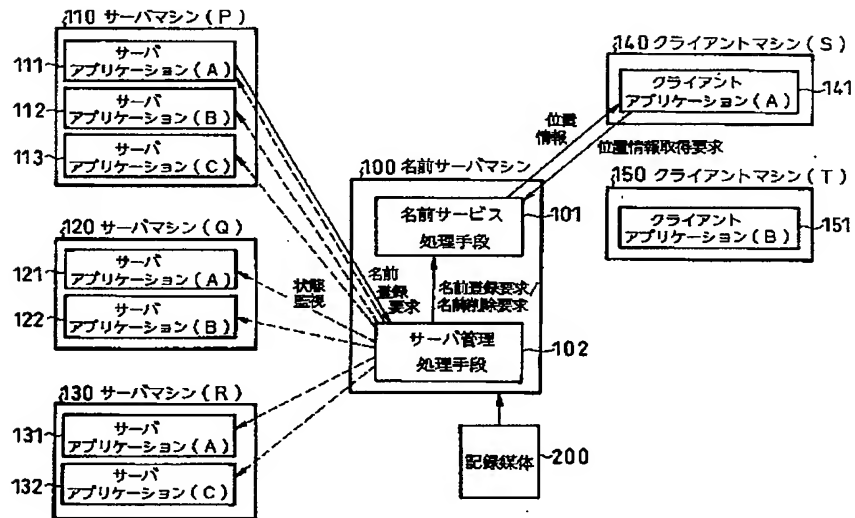
【図 10】



【図 9】



【図 11】



# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-209568

(43)Date of publication of application : 03.08.2001

(51)Int.Cl. G06F 12/00  
G06F 13/00  
// H04N 7/173

(21)Application number : 2000-024769

(71)Applicant : JISEDAL JOHO HOSO  
SYSTEM KENKYUSHO:KK  
SONY CORP

(22)Date of filing : 28.01.2000

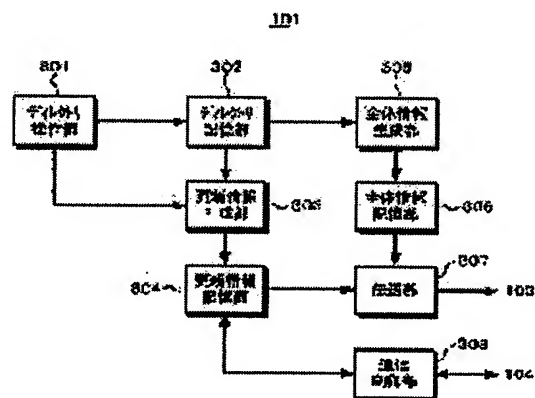
(72)Inventor : NISHIO IKUHIKO  
GONNO YOSHIHISA  
HARAOKA KAZUO  
TAKABAYASHI KAZUHIKO  
YAMAGISHI YASUAKI

## (54) TRANSMITTER, RECEIVER, TRANSMITTER-RECEIVER, METHODS FOR TRANSMISSION AND RECEPTION

### (57)Abstract:

PROBLEM TO BE SOLVED: To reduce processing load for receiving in directory service by synchronized approach to total information.

SOLUTION: Information for directory renewal operation is supplied to updating information generation part 303, and updating information for reflecting updating contents to a directory server is generated. The renewed information is then stored to a renewal storage part 304 and transmitted to a network 103 of the information. The total directory information is periodically generated in a part 305 for generating the total information, stored in a total information storage 306 and transmitted to the information network 103. Version number that happens to be renewed immediately below the container is stored in the respective container entries to be stored in a directory storage 302. In addition to the up-to-date contents to be processed, the container



entry name gained at renewal and the version number finally renewed immediately below the container entry are involved in the total information to be created in an area 305 where the total information is generated.



(5)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号  
特開2001-209568  
(P2001-209568A)

(43) 公開日 平成13年8月3日(2001.8.3)

(51) Int.Cl. <sup>7</sup>	識別記号	F I	テ-マコード*(参考)
G 0 6 F 12/00	5 4 5	G 0 6 F 12/00	5 4 5 A 5 B 0 8 2
	5 1 7		5 1 7 5 B 0 8 9
13/00	3 5 7	13/00	3 5 7 Z 5 C 0 6 4
// H 0 4 N 7/173	6 1 0	H 0 4 N 7/173	6 1 0 Z

審査請求 未請求 請求項の数10 O L (全 13 頁)

(21) 出願番号 特願2000-24769(P2000-24769)

(22) 出願日 平成12年1月28日(2000.1.28)

(71) 出願人 597136766

株式会社次世代情報放送システム研究所  
東京都台東区西浅草1丁目1-1

(71) 出願人 000002185

ソニー株式会社  
東京都品川区北品川6丁目7番35号

(72) 発明者 西尾 郁彦

東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

(74) 代理人 100082762

弁理士 杉浦 正知

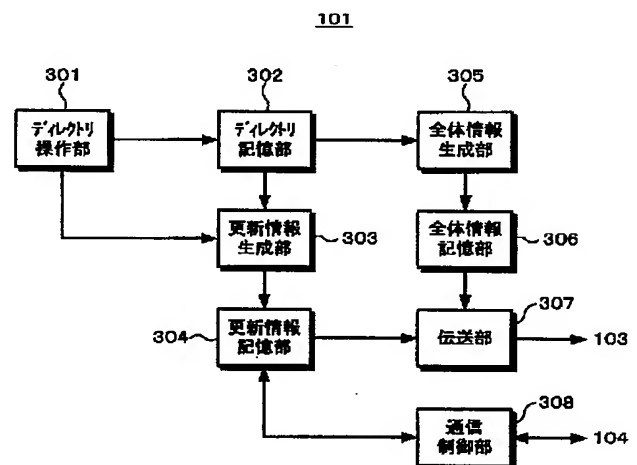
最終頁に続く

(54) 【発明の名称】 送信装置、受信装置、送受信装置、送信方法および受信方法

(57) 【要約】

【課題】 ディレクトリサービスにおいて、全体情報で同期をとる時に、受信側の処理の負荷を軽くする。

【解決手段】 ディレクトリの更新操作の情報が更新情報生成部303に供給され、更新内容を受信側ディレクトリサーバに反映するための更新情報が生成される。更新情報が更新情報記憶部304に記憶され、また、同報ネットワーク103に送信される。全体情報生成部305では、ディレクトリ全体の情報が定期的に生成され、全体情報が全体情報記憶部306に記憶され、同報ネットワーク103に送信される。ディレクトリ記憶部302に記憶される各コンテナエントリには、そのコンテナエントリの直下のエントリに更新が起こったときのバージョン番号を格納する。全体情報生成部305が生成する全体情報には、更新の起こったコンテナエントリ名と更新操作内容に加えて、最後にそのコンテナエントリ直下に更新が起きた時のバージョン番号を含める。



**【特許請求の範囲】**

**【請求項 1】** 木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、

上記ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに变化するバージョン番号を管理するディレクトリ操作手段と、

上記ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、

上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、

上記全体情報および上記更新情報をネットワークにより送信する伝送手段とを備え、

上記全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信装置。

**【請求項 2】** 請求項 1 において、

上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持つことを特徴とする送信装置。

**【請求項 3】** 請求項 1 において、

上記ネットワークが同報ネットワークであることを特徴とする送信装置。

**【請求項 4】** 請求項 1 において、

上記ネットワークが双方向ネットワークであることを特徴とする送信装置。

**【請求項 5】** 木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、

上記ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、

上記ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信する受信手段と、

上記更新情報および上記全体情報の内容に従ってディレクトリを更新する更新手段とを備え、

上記全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、

上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、

上記更新手段は、受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信装置。

**【請求項 6】** 請求項 5 において、

上記ネットワークが同報ネットワークであることを特徴とする送信装置。

**【請求項 7】** 請求項 5 において、

上記ネットワークが双方向ネットワークであることを特徴とする送信装置。

**【請求項 8】** 送信装置と受信装置とがネットワークを介して接続され、上記送信装置および受信装置の間で、ディレクトリの同期をとるようにした送受信装置において、

上記送信装置は、

木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、

上記ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに变化するバージョン番号を管理するディレクトリ操作手段と、

上記ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、

上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、

上記全体情報および上記更新情報をネットワークにより送信する伝送手段とを備え、

上記全体情報生成手段は、全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成するように構成され、

上記受信装置は、

木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、

上記ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、

上記送信装置から上記更新情報および上記全体情報を上記ネットワークを介して受信する受信手段と、

上記更新情報および上記全体情報の内容に従ってディレクトリを更新する更新手段とを備え、

上記更新手段は、受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映するように構成されることを特徴とする送受信装置。

**【請求項 9】** ディレクトリ記憶手段に記憶され、木構造のデータ構造をもったディレクトリを更新すると共に、上記ディレクトリに更新を行うたびに变化するバージョン番号を管理するステップと、

上記ディレクトリの更新内容を表す更新情報を生成するステップと、

上記ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成するステップと、  
上記全体情報および上記更新情報をネットワークにより送信するステップとからなり、

上記全体情報の生成に際して、エントリ名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信方法。

【請求項 10】 ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信するステップと、

上記更新情報および上記全体情報の内容に従ってディレクトリを更新するステップとを備え、

上記全体情報は、エントリ名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、

上記ディレクトリ記憶手段に記憶されたディレクトリの各エントリは、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、

受信した上記全体情報が持つバージョン番号と、上記ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な上記全体情報を判別し、必要な上記全体情報のみを上記ディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 この発明は、例えば不特定多数へのデータ配信の技術分野で用いられ、配信されたデータを効率良く更新管理するための送信装置、受信装置、送受信装置、送信方法および受信方法に関する。

【0002】

【従来の技術】 ディレクトリサービスの規格として、X. 500やLDAP (Lightweight Directory Access Protocol) が知られている。X. 500は、ITU-T (International Telecommunication Union-Telecommunication Standardization Sector)で勧告されているディレクトリサービスの国際標準であり、ディレクトリの概念や、階層構造、サービス、オブジェクトの定義などが規定されている。

【0003】 X. 500は、汎用性の高いディレクトリサービスプロトコルである反面、実装が重くなる仕様となっており、その実装コストが高い。このため、これを、インターネット上のWWW (World Wide Web) ブラウザや、電子メールソフトウェアなどからも簡単に利用することができるように、軽量化、簡素化したサービスプロトコルとして、LDAPが開発された。

【0004】 LDAPは、サーバ・クライアント型のモデルで構成されており、ディレクトリ情報を格納するサ

ーバ (ディレクトリサーバ) と、それにアクセスするクライアントとの間のリクエスト・レスポンス型のメッセージ転送プロトコルを定義している。ディレクトリサーバで集中管理されるディレクトリ情報の規模が大きくなると、クライアントのアクセスが集中することによって、サーバの質が低下するために、複数のディレクトリサーバによって、ディレクトリ情報が分散管理される。

【0005】 ディレクトリ情報が分散管理されている場合、あるディレクトリサーバが管理しているエントリ

(ディレクトリ情報の最小管理単位) の複製が他の1以上のディレクトリサーバに存在することになる。したがって、あるディレクトリサーバにおいてエントリの更新が生じた時には、他のディレクトリサーバにおいても、複製を更新する必要がある。このように、エントリの内容が異なるものとならないように、同期管理が必要とされる。すなわち、あるディレクトリサーバが管理しているエントリが更新された場合に、他のディレクトリサーバが管理しているそのエントリの複製の更新を即座に行うことが必要とされる。同期管理方式として、同報ネットワークを使用した放送型プロトコルを主体としたものが先に提案されている (特願平 10-277352 号および特願平 10-277353 号参照)。

【0006】

【発明が解決しようとする課題】 上述した同期管理方式を有するディレクトリサービスシステムにおいて、更新内容は、更新前と更新後の差分のデータとして伝送される。若し、受信側のディレクトリサーバが電源オフ等のために長い間休止していると、連続した差分更新通知メッセージを取りこぼすおそれが高くなる。このような場合、放送によって伝送される差分情報だけでは、受信側のディレクトリを同期させることができなくなる。その結果、受信側は、差分情報ではなく、既に受信しているディレクトリ情報を捨て、ディレクトリ全体の情報を最初から受信してディレクトリ全体を再構築することによって、同期した状態へ回復することになる。一般的にディレクトリ全体の情報は、差分に比べて非常に大きなデータサイズとなるため、ディレクトリ回復作業が受信側にとって負荷が高い処理となる。

【0007】 従って、この発明の目的は、全体の情報を受信することによって受信側が送信側とディレクトリを同期させる際に、既に受信済みのディレクトリ構造を可能な限り保持しながらディレクトリ同期を行い、それによって処理の負荷を軽減することが可能な送信装置、受信装置、送受信装置、送信方法および受信方法を提供することにある。

【0008】

【課題を解決するための手段】 上述した課題を解決するために、請求項 1 の発明は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新す

ると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、全体情報および更新情報をネットワークにより送信する伝送手段とを備え、全体情報生成手段は、全体情報の生成に際して、エン트리名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信装置である。

【0009】請求項5の発明は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信する受信手段と、更新情報および全体情報の内容に従ってディレクトリを更新する更新手段とを備え、全体情報は、エン트리名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、ディレクトリ記憶手段に記憶されたディレクトリの各エント리는、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、更新手段は、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信装置である。

【0010】請求項8の発明は、送信装置と受信装置とがネットワークを介して接続され、送信装置および受信装置の間で、ディレクトリの同期をとるようにした送受信装置において、送信装置は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されたディレクトリを更新すると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するディレクトリ操作手段と、ディレクトリ操作手段によるディレクトリの更新内容を表す更新情報を生成する更新情報生成手段と、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成する全体情報生成手段と、全体情報および更新情報をネットワークにより送信する伝送手段とを備え、全体情報生成手段は、全体情報の生成に際して、エン트리名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成するように構成され、受信装置は、木構造のデータ構造をもったディレクトリを記憶するディレクトリ記憶手段と、ディレクトリ記憶手段に記憶されるディレクトリを検索、閲覧するためのディレクトリ操作手段と、送

信装置から更新情報および全体情報をネットワークを介して受信する受信手段と、更新情報および全体情報の内容に従ってディレクトリを更新する更新手段とを備え、更新手段は、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映するように構成されることを特徴とする送受信装置である。

【0011】請求項9の発明は、ディレクトリ記憶手段に記憶され、木構造のデータ構造をもったディレクトリを更新すると共に、ディレクトリに更新を行うたびに変化するバージョン番号を管理するステップと、ディレクトリの更新内容を表す更新情報を生成するステップと、ディレクトリ記憶手段に記憶されたディレクトリの内容全体を表す全体情報を生成するステップと、全体情報および更新情報をネットワークにより送信するステップとからなり、全体情報の生成に際して、エン트리名と、更新操作と、そのエントリの直下に対して更新が起きた時のバージョン番号とを含む全体情報を生成することを特徴とする送信方法である。

【0012】請求項10の発明は、ディレクトリ記憶手段に記憶されたディレクトリを更新するための更新情報およびディレクトリ全体の全体情報をネットワークを介して受信するステップと、更新情報および全体情報の内容に従ってディレクトリを更新するステップとを備え、全体情報は、エン트리名と、そのエントリの直下に対して更新が起きた時のバージョン番号と、更新操作とを含み、ディレクトリ記憶手段に記憶されたディレクトリの各エント리는、そのエントリの直下に対して更新が起きた時のバージョン番号を持ち、受信した全体情報が持つバージョン番号と、ディレクトリ記憶手段に記憶されたディレクトリのバージョン番号とを比較することによって、必要な全体情報を判別し、必要な全体情報のみをディレクトリ記憶手段に記憶されたディレクトリに反映することを特徴とする受信方法である。

【0013】この発明では、各エン트리に対して、その直下に更新が起きた時のバージョン番号を付加することによって、全体情報によってディレクトリの同期を取り直す時に、必要な部分のみを処理することができる。

【0014】

【発明の実施の形態】以下、この発明の一実施形態について説明する。図1は、一実施形態におけるデータ配信システムの一例を示す。送信側ディレクトリサーバ101は、ディレクトリを保持、管理している。ディレクトリは、木構造のデータ構造を有し、その最小管理単位がコンテナエン트리（単にエン트리とも言う）であり、また、コンテナエントリの追加、削除を行うことができる。一例として、ディレクトリサーバ101には、情報提供者（図示しない）から提供された交通情報、天気情

報、株価情報等のリアルタイムで変化するデータ、リアルタイムで変化しないデータ、テキストデータ、画像データ、オーディオデータ、コンピュータプログラム等のデータが記憶される。ディレクトリのひとかたまりのデータがコンテンツ、またはオブジェクトと呼ばれる。また、送信側ディレクトリサーバ101の一例は、放送局である。

【0015】送信側ディレクトリサーバ101は、同報ネットワーク（例えば放送ネットワーク）103および双方向ネットワーク104を介して受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>にディレクトリを提供する。双方向ネットワーク104は、具体的には、アナログ公衆電話網、ISDN(Integrated Services Digital Network)、インターネット等である。受信側ディレクトリサーバの個数は、実際には、無数あるが、図1は、代表的に2個の受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>を示している。

【0016】受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>は、送信側ディレクトリサーバ101から提供されるディレクトリを受信し、蓄積し、利用者に提供するものである。そのため、送信側ディレクトリサーバ101に記憶されるディレクトリの全て、またはその一部を同報ネットワーク103および双方向ネットワーク104を介して受信し、記憶（蓄積）する。

【0017】また、送信側ディレクトリサーバ101において起こったディレクトリの更新は、同報ネットワーク103および双方向ネットワーク104を介して受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>に反映される。それによって、受信側ディレクトリサーバ101と送信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>の間でディレクトリの内容の同期が取られる。

【0018】ディレクトリは、図2に示すように、木構造で管理されるデータである。木構造を構成する各ノードのうちさらにその下にノードをもつものがコンテナエントリ201であり、一番下に位置するノードがリーフノード202である。図2中、C1、C2、・・・、C4がコンテナエントリであり、L1、L2、L3がリーフノードである。例えば放送番組表の場合では、コンテナエントリC2がニュース番組であり、リーフノードL1が午後9時に放送されるニュース番組である。勿論、職業別電話帳等も図2に示す木構造で表すことができる。

【0019】図3は、送信側ディレクトリサーバ101の構成例を示す。ディレクトリ記憶部302には、図2に示す構造を持ったディレクトリが記憶される。ディレクトリ記憶部302にデータを追加したり、記憶されたデータを削除するのがディレクトリ操作部301である。ディレクトリ操作部301によるディレクトリの更新操作の情報が更新情報生成部303に供給される。更新操作の情報を受けて、更新情報生成部303では、デ

ィレクトリの更新内容を受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>のディレクトリに反映するための更新情報が生成される。

【0020】更新情報生成部303によって生成された更新情報が更新情報記憶部304に記憶される。更新情報記憶部304には、受信側での更新情報の取りこぼしを想定して一定期間繰り返して更新情報を送信するために、複数の更新情報が記憶される。更新情報記憶部304に記憶された更新情報は、伝送部307によって同報ネットワーク103に繰り返して送信される。

【0021】ディレクトリ記憶部302には、全体情報生成部305が接続される。全体情報生成部305では、受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>での更新情報の取りこぼしを考慮し、ディレクトリ記憶部302に記憶されたディレクトリ全体の情報が定期的に生成される。生成された全体情報が全体情報記憶部306に記憶され、伝送部307から同報ネットワーク103に一定期間、繰り返して送信される。

【0022】双方向ネットワーク104に通信制御部308が接続される。双方向ネットワーク104を介して受信された受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>からの要求に従って、更新情報記憶部304に記憶された更新情報またはディレクトリ記憶部302に記憶されるディレクトリ全体を双方向ネットワーク104を介して受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>に提供する。

【0023】図4は、受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>の構成例を示す。同報ネットワーク103に対して受信部401が接続される。受信部401が同報ネットワーク103を介して送信側ディレクトリサーバ101からディレクトリの更新情報および全体情報を受信する。受信した情報は、更新操作部403において解釈され、ディレクトリ記憶部404に記憶されているディレクトリに反映される。全体情報を受信したのであれば、そのデータをディレクトリ記憶部404に記憶する。また、更新情報を受信したのであれば、ディレクトリ記憶部404に記憶されているディレクトリの指示された部分を更新する。

【0024】双方向ネットワーク104に対して通信制御部402が接続される。通信制御部402は、受信部401で取りこぼした全体情報または更新情報を送信側ディレクトリサーバ101に要求する。そして、受信した全体情報や更新情報を更新操作部403を通じてディレクトリ記憶部404に記憶されているディレクトリに反映する。ディレクトリ操作部405は、ディレクトリ記憶部404に記憶されているディレクトリを閲覧したり、参照するための操作を行うためのものである。

【0025】図5を参照して、送信側ディレクトリサーバ101におけるディレクトリの更新例について説明する。更新操作は、ディレクトリ操作部301によって行

われる。まず、コンテナエントリC0にC1、C2が順に追加される(データ構造501)。次に、コンテナエントリC3がコンテナエントリC1の下に追加される

(データ構造502)。コンテナエントリC2の下にコンテナエントリC4が追加される(データ構造503)。コンテナエントリC2の下にコンテナエントリC5が追加される(データ構造504)。最後にコンテナエントリC2の下からコンテナエントリC4が削除される(データ構造505)。

【0026】図5に示すようなディレクトリの更新が起こった場合を考えると、送信側ディレクトリサーバ101の更新情報生成部303および全体情報生成部305では、図6に示すような全体情報および更新情報が生成される。これらの全体情報および更新情報が同報ネットワーク103を介して送信される。また、受信側ディレクトリサーバ102<sub>1</sub>、102<sub>2</sub>の要求に応じて、要求された全体情報または更新情報が双方向ネットワーク104を介して送信される。

【0027】一実施形態においては、ディレクトリ記憶部302に記憶される各コンテナエントリには、そのコンテナエントリの直下のエントリに更新が起こったときのスキーマのバージョン(バージョン番号と呼ぶことにする)を格納する。すなわち、コンテナエントリの状態は、(コンテナエントリ名、バージョン番号)で表すことができる。バージョン番号は、更新のたびに変化し、更新の順序を示す値であれば、昇順の数、降順の数等、種々の値を使用できる。

【0028】全体情報生成部305が生成する全体情報には、更新の起こったコンテナエントリ名と更新操作内容に加えて、最後にそのコンテナエントリ直下に更新が起きた時のバージョン番号を含める。つまり、全体情報は、(バージョン番号、コンテナエントリ名、更新操作)で表される。更新情報も同様に表す。操作は、コンテナエントリの追加、または削除である。

【0029】図6中の全体情報について説明すると、最初のバージョン番号(0.3)がデータ構造501に対応したものである。すなわち、仮想的なコンテナエントリ(またはノード)rにコンテナエントリC0を追加したものがバージョン番号(0.1)の全体情報(0.1, r, +C0)である。バージョン番号(0.1)に対して、コンテナエントリC0にC1を追加したバージョン番号が(0.2)である。さらに、コンテナエントリC2を追加した全体情報が(0.3, C0, +C1)である。

【0030】コンテナエントリC0の直下に最後に更新が起きた時のバージョン番号は、コンテナエントリC2が追加された時のバージョン番号(0.3)であるので、コンテナエントリC1を追加した時の全体情報が(0.3, C0, +C1)とされる。以上の3個の全体情報によって、データ構造501が復元できる。また、

更新情報としての差分情報は、バージョン番号(0.2)からバージョン番号(0.3)に変化した時の変化を示すように、コンテナエントリC0の直下に最後に更新が起きた時のバージョン番号(0.3)と、更新操作からなる。すなわち、(0.3, C0, +C2)である。(+)が追加を表す。

【0031】次に、コンテナエントリC1に対して、コンテナエントリC3を追加してデータ構造502(バージョン番号(1.0))が生成される。この場合には、コンテナエントリC1の直下に最後に更新が起きた時のバージョン番号が(1.0)であるので、(1.0, C1, +C3)の全体情報が生成される。コンテナエントリC0についてのバージョン番号は、(0.3)のままである。また、差分情報が(1.0, C1, +C3)となる。

【0032】次に、コンテナエントリC2に対して、コンテナエントリC4を追加してデータ構造503(バージョン番号(1.1))が生成される。この場合には、コンテナエントリC2の直下に最後に更新が起きた時のバージョン番号が(1.1)であるので、(1.1, C2, +C4)の全体情報が生成される。コンテナエントリC0についてのバージョン番号は、(0.3)のまま、コンテナエントリC1についてのバージョン番号は、(1.0)のままである。また、差分情報が(1.1, C2, +C4)となる。

【0033】次に、コンテナエントリC2に対して、コンテナエントリC5を追加してデータ構造504(バージョン番号(1.2))が生成される。この場合には、コンテナエントリC2の直下に最後に更新が起きた時のバージョン番号が(1.2)であるので、(1.2, C2, +C5)の全体情報が生成される。これと共に、バージョン番号(1.1)では、(1.1, C2, +C4)であった全体情報が(1.2, C2, +C4)に変更される。また、差分情報が(1.2, C2, +C5)となる。

【0034】最後に、コンテナエントリC2に対して追加されたコンテナエントリC4が削除されてデータ構造505(バージョン番号(1.3))が生成される。この場合には、コンテナエントリC2の直下に最後に更新が起きた時のバージョン番号が(1.3)であるので、(1.3, C2, +C5)の全体情報が生成される。これと共に、コンテナエントリC4の削除の操作と対応して、全体情報(1.2, C2, +C4)が削除される。また、差分情報が(1.3, C2, -C4)となる。

(-)が削除を表す。

【0035】図7は、送信側ディレクトリサーバ101において、更新操作がなされた時になされる各コンテナエントリCnのバージョン番号管理の流れを示すフローチャートである。ステップS1において、あるコンテナエントリCnの直下に更新(コンテナエントリの追加ま

たは削除)が起こると、ステップS2において、バージョン番号Vを1増加する。そして、ステップS3において、+1で得られたバージョン番号VをコンテナエントリC<sub>n</sub>の直下に更新が起きた時のバージョン番号C<sub>n</sub>vとして記憶する。

【0036】図8は、送信側ディレクトリサーバ101における全体情報の生成処理の流れを示すフローチャートである。コンテナエントリの総数をNとし、処理の対象とするコンテナエントリがi番目のコンテナエントリC<sub>i</sub>とすると、ステップS11では、(i<N)かどうか決定される。(i<N)の関係が成立するときは、全てのコンテナエントリの処理が終了していないことを意味するので、処理がステップS12に移る。若し、(i<N)の関係が成立しないときは、全てのコンテナエントリに関する処理終了したものと決定され、処理が終了する。

【0037】ステップS12においては、コンテナエントリC<sub>i</sub>の直下にある全てのコンテナエントリC<sub>j</sub>に関して処理がなされる。すなわち、ステップS13において、全体情報に付加するバージョン番号vをコンテナエントリC<sub>i</sub>の記憶しているバージョン番号C<sub>i</sub>vとし(ステップS13)、次のステップS14において、全体情報(v, C<sub>i</sub>, +C<sub>j</sub>)、すなわち、(C<sub>i</sub>v, C<sub>i</sub>, +C<sub>j</sub>)を生成する。

【0038】全てのコンテナエントリC<sub>j</sub>についての処理(ステップS13およびステップS14)が終了すると、ステップS15に処理が移り、コンテナエントリ番号iが1増加される。そして、ステップS11に戻り、全てのコンテナエントリC<sub>i</sub>についての処理が終了したか否かが決定される。

【0039】ここで、受信側ディレクトリサーバ102<sub>1</sub>, 102<sub>2</sub>が長期間、休止状態であった後に復帰した場合、ディレクトリ記憶部404に記憶されている情報を最新の状態に同期させる既存の方法について説明する。まず、ディレクトリの全体情報を受信する。ディレクトリの全体情報を受信し、ディレクトリを同期する際に、既にディレクトリ記憶部404に記憶されている情報を全て破棄し、改めて全体情報からディレクトリを再構築する。しかしながら、このような既存の同期方法は、受信側ディレクトリサーバにとって、処理の負荷が重い。

【0040】この発明では、ディレクトリを送信側ディレクトリサーバと同期させるために、既に受信済の情報に関して変化が起きたか否かを先に判定することによって処理の負荷を軽減することが可能なものである。その判定のために、送信側ディレクトリサーバ101において、上述したように生成された全体情報中のバージョン番号、すなわち、各コンテナエントリに付加されている、そのコンテナエントリの直下のエントリに更新が起こったときのバージョン番号を使用する。

【0041】図9は、受信側ディレクトリサーバ102<sub>1</sub>, 102<sub>2</sub>の更新処理の流れを示すフローチャートである。最初のステップS21において、最新の差分更新情報に含まれるバージョン番号V<sub>d</sub>と受信側に記憶されているバージョン番号V<sub>s</sub>とを比較する。(V<sub>d</sub>=V<sub>s</sub>+1)ならば、取りこぼしがなく、正常に差分更新情報を受信していることが分かる。その場合には、ステップS22において、差分更新を適用し、バージョン番号V<sub>s</sub>を1増加し、更新処理を終了する。ステップS22の処理によって、受信側のバージョン番号が送信側のバージョン番号と等しくなる。

【0042】ステップS21において、(V<sub>d</sub>=V<sub>s</sub>+1)でなければ、すなわち、(V<sub>d</sub>>V<sub>s</sub>+1)ならば、差分更新情報の取りこぼしが起きているものと決定される。この場合では、全体情報を受信することによって、復旧が図られる。まず、ステップS23において、全ての全体情報を受信したか否かが決定される。全ての全体情報を受信していれば、復旧が完了する。

【0043】ステップS23において、全ての全体情報を受信していないならば、ステップS24において、全体情報(v, C<sub>i</sub>, +C<sub>j</sub>)を受信する。ステップS25では、全体情報中のバージョン番号vと受信側に記憶されているバージョン番号V<sub>s</sub>との関係が(v>V<sub>s</sub>)か否かが決定される。(v>V<sub>s</sub>)でなければ、処理がステップS23に戻る。(v>V<sub>s</sub>)であれば、コンテナエントリC<sub>i</sub>の直下に不整合が起きていることが分かるので、ステップS26において、コンテナエントリC<sub>i</sub>の直下の全てのコンテナエントリが削除済か否かが決定される。これは、不整合によってコンテナエントリC<sub>i</sub>の直下のコンテナエントリが更新を正しく反映していないおそれがあるからである。

【0044】削除済でないならば、ステップS27において、コンテナエントリC<sub>i</sub>の直下のコンテナエントリを全て削除し、ステップS28に処理が移る。ステップS26において、削除済と決定されたときも、処理がステップS28に移る。ステップS28では、全体情報(v, C<sub>i</sub>, +C<sub>j</sub>)の内容をコンテナエントリC<sub>i</sub>の直下に反映するために、コンテナエントリC<sub>i</sub>にコンテナエントリC<sub>j</sub>を追加する。そして、処理がステップS23に戻る。全ての全体情報を受信するまで、ステップS24~S28の処理が繰り返される。

【0045】図6の例を参照してより具体的に説明すると、例えば受信側ディレクトリサーバがバージョン番号(1. 1)の状態(全体情報(1. 1, C2, +C4))の後、バージョン番号(1. 2)の間、電源切断等の何らかの理由によって休止状態となったものとする。

【0046】この場合には、全体情報(1. 2, C2, +C4)を受信すると、ステップS25(図9)の関係(v>V<sub>s</sub>)が満たされ、コンテナエントリC2の直下



のエントリ C 4 が削除され（ステップ S 2 7）、C 2 の直下に C 4 が追加される（ステップ S 2 8）。そして、次の全体情報（1. 2, C 2, +C 5）を受信すると、コンテナエントリ C 5 が C 2 に追加される。このようにして、コンテナエントリ C 2 の部分のみを処理することで、ディレクトリの同期なしうる。

【0047】この発明を適用しない場合、受信側ディレクトリサーバ 102<sub>1</sub>, 102<sub>2</sub> は、ディレクトリ記憶部 404 に記憶されるバージョン番号（1. 1）のディレクトリ構造を全て破棄し、バージョン（1. 3）の全体情報を受信してディレクトリの同期をとることになる。しかしながら、実際には、バージョン（1. 1）と（1. 3）の間のディレクトリの違いは、C 4、C 5 の部分のみであり、その他の部分については同期処理を必要としない。この発明によれば、コンテナエントリ C 2 以下の部分についてのみ同期処理を行うことになるので、受信側の処理の負荷を軽くすることができる。

【0048】この発明は、上述した実施形態等に限定されるものではなく、この発明の要旨を逸脱しない範囲内で様々な変形や応用が可能である。

【0049】

【発明の効果】この発明では、ディレクトリの更新において、各コンテナエントリに対して、その直下に更新処理が起きたときのバージョン番号を付加することにより、全体情報によって同期を取り直す時に、必要な部分のみを処理することが可能となり、受信側の処理の負荷を軽くすることができる。

【図面の簡単な説明】

【図 1】この発明の一実施形態である、ディレクトリサービスシステムの構成の一例を示す略線図である。

【図 2】この発明の一実施形態におけるディレクトリの木構造を説明するための略線図である。

【図 3】この発明の一実施形態における送信側ディレクトリサーバの一例を示すブロック図である。

【図 4】この発明の一実施形態における受信側ディレクトリサーバの一例を示すブロック図である。

【図 5】ディレクトリエントリの更新例を順に示す略線図である。

【図 6】受信側ディレクトリサーバにおけるディレクトリ更新処理の一例における全体情報および更新情報を示す略線図である。

【図 7】この発明の一実施形態における、送信側でのコンテナエントリの更新バージョン番号管理の処理を示すフローチャートである。

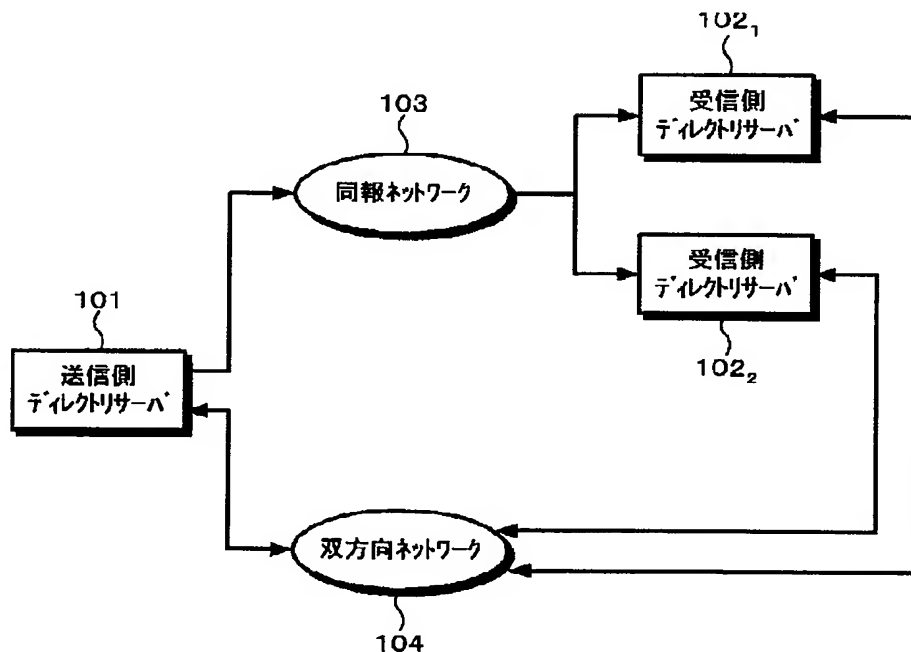
【図 8】この発明の一実施形態における、送信側での全体情報の生成処理を示すフローチャートである。

【図 9】この発明の一実施形態における、受信側でのディレクトリの更新処理を示すフローチャートである。

【符号の説明】

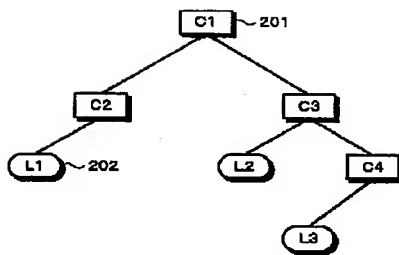
101・・・送信側ディレクトリサーバ、102<sub>1</sub>, 102<sub>2</sub>・・・受信側ディレクトリサーバ、103・・・同報ネットワーク、201・・・コンテナエントリ、202・・・リーフノード、302・・・ディレクトリ記憶部、303・・・更新情報生成部、304・・・更新情報記憶部、401・・・受信部、404・・・ディレクトリ記憶部

【図 1】

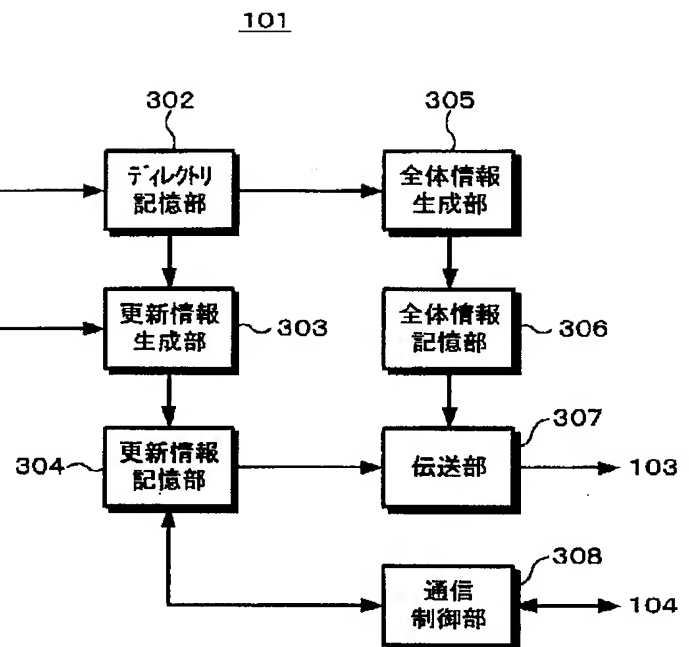




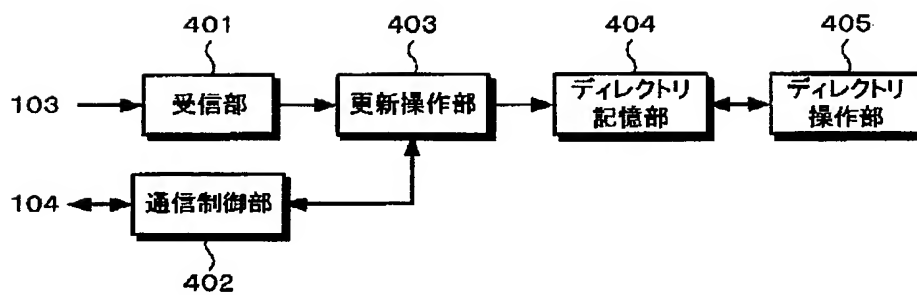
【図2】



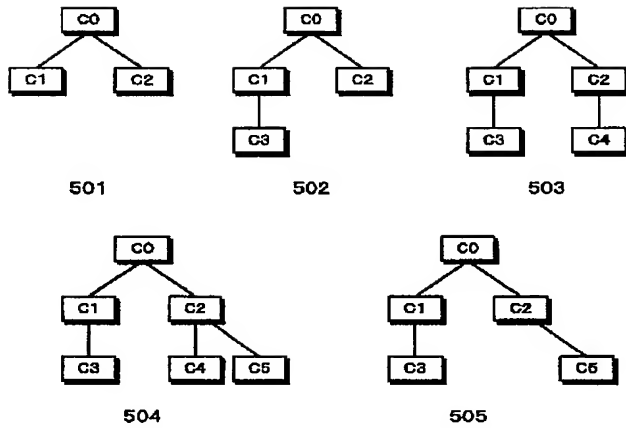
【図3】



【図4】

102<sub>1</sub>, 102<sub>2</sub>

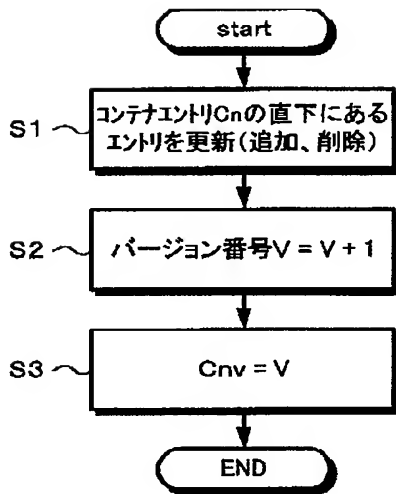
【図 5】



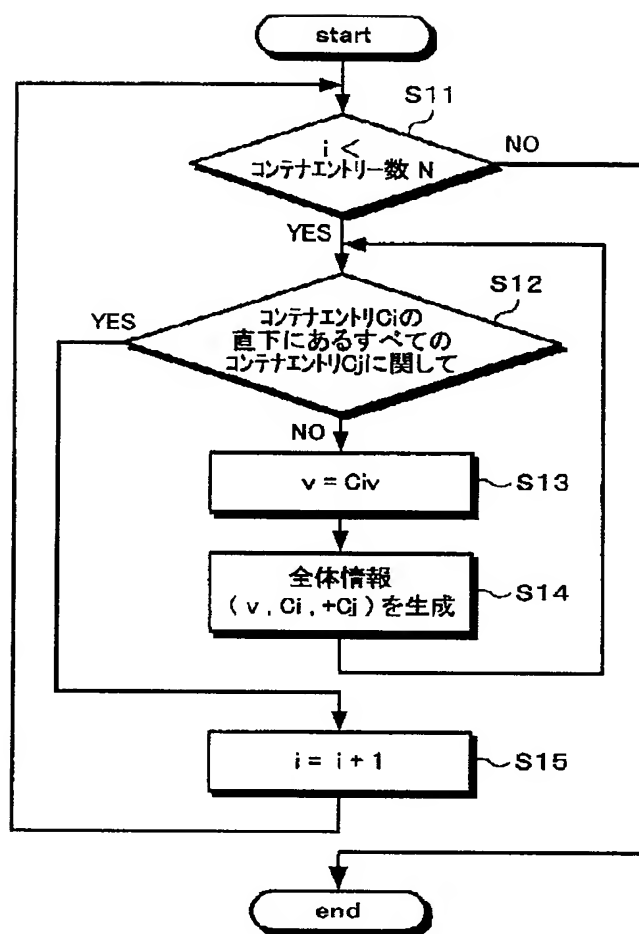
【図 6】

ver.	全体情報	差分情報
0.3	$(0.1, r, +C0)$ $(0.3, C0, +C1)$ $(0.3, C0, +C2)$	$(0.3, C0, +C2)$
1.0	$(0.1, r, +C0)$ $(0.3, C0, +C1)$ $(0.3, C0, +C2)$ $(1.0, C1, +C3)$	$(1.0, C1, +C3)$
1.1	$(0.1, r, +C0)$ $(0.3, C0, +C1)$ $(0.3, C0, +C2)$ $(1.0, C1, +C3)$ $(1.1, C2, +C4)$	$(1.1, C2, +C4)$
1.2	$(0.1, r, +C0)$ $(0.3, C0, +C1)$ $(0.3, C0, +C2)$ $(1.0, C1, +C3)$ $(1.2, C2, +C4)$ $(1.2, C2, +C5)$	$(1.2, C2, +C5)$
1.3	$(0.1, r, +C0)$ $(0.3, C0, +C1)$ $(0.3, C0, +C2)$ $(1.0, C1, +C3)$ $(1.3, C2, +C5)$	$(1.3, C2, -C4)$

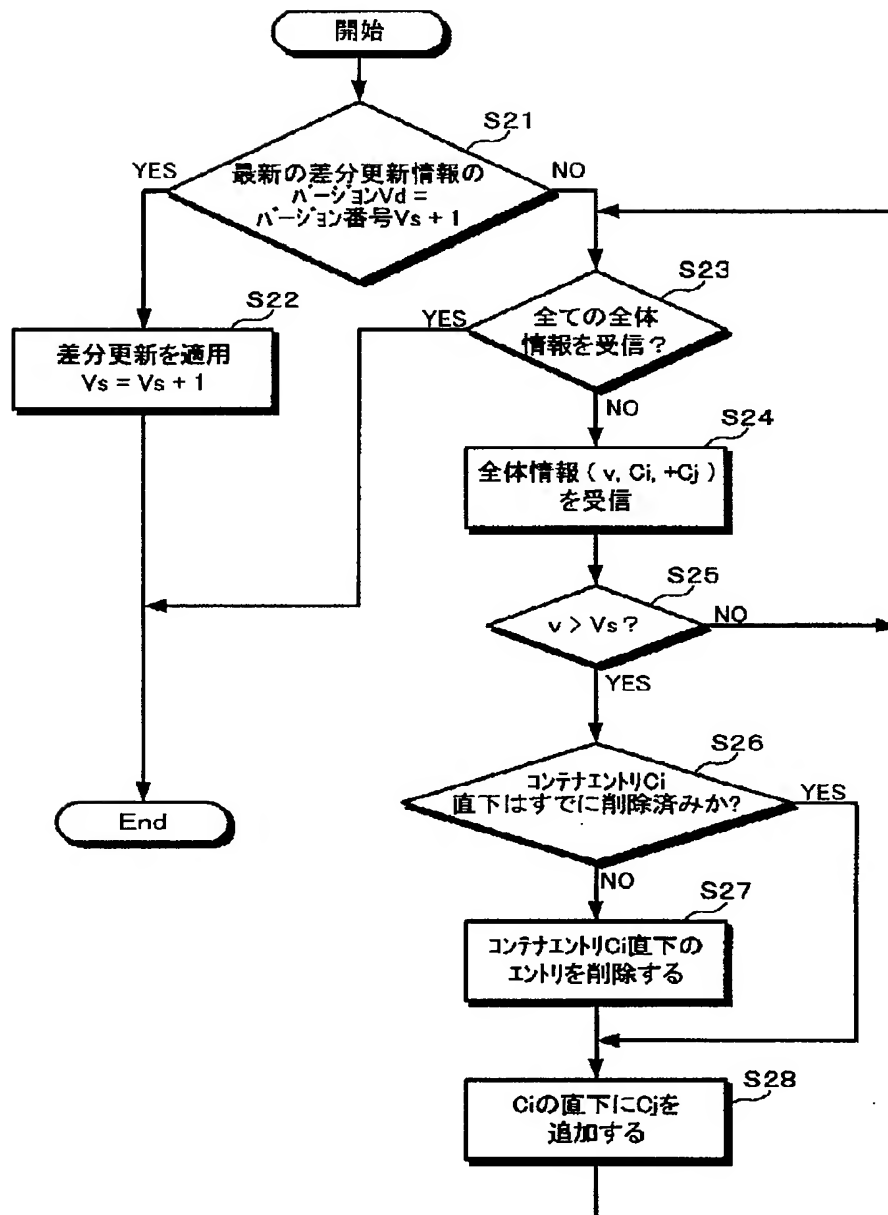
【図 7】



【図 8】



【図9】



フロントページの続き

(72)発明者 権野 善久  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

(72)発明者 原岡 和生  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

(72)発明者 高林 和彦  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

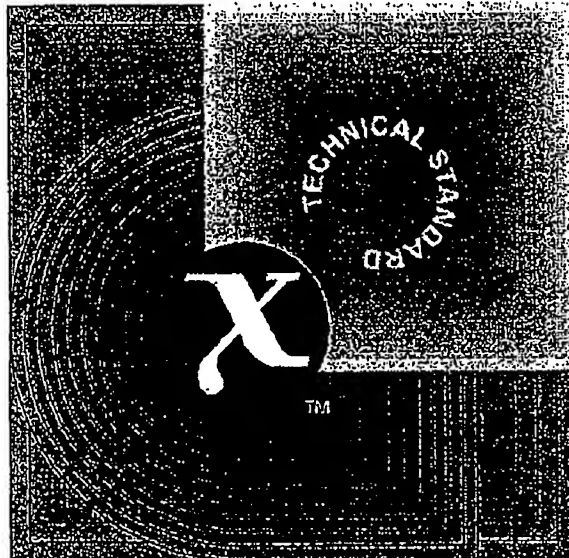
(72)発明者 山岸 靖明  
東京都品川区北品川6丁目7番35号 ソニ  
ー株式会社内

F ターム (参考) 5B082 FA00 GA15  
5B089 GA11 JA11 JB22 KA06 KB10  
KC30 KE07  
5C064 BB05 BC06 BC10 BC16 BC20  
BD01 BD07

## Technical Standard

---

### Federated Naming: The XFN Specification



THE *Open* GROUP

USPTO

[This page intentionally left blank]

# *X/Open CAE Specification*

**Federated Naming:  
The XFN Specification**

*X/Open Company Ltd.*



**© July 1995, X/Open Company Limited**

**All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the copyright owners.**

**X/Open CAE Specification**

**Federated Naming: The XFN Specification**

**ISBN: 1-85912-052-0**

**X/Open Document Number: C403**

**Published by X/Open Company Ltd., U.K.**

**Any comments relating to the material contained in this document may be submitted to X/Open at:**

**X/Open Company Limited  
Apex Plaza  
Forbury Road  
Reading  
Berkshire, RG1 1AX  
United Kingdom**

**or by Electronic Mail to:**

**XoSpecs@xopen.org**

## 2.3 XFN Usage and Implementation Models

This section describes how the XFN interface can be used and implemented. The usage model is from a client's perspective. The implementation model is from a service provider's perspective.

### 2.3.1 Basic Usage Model

Most clients of the XFN interface will only be interested in lookups. Their usage of the XFN interface will amount essentially to:

- obtaining the Initial Context
- looking up one or more names relative to the Initial Context.

Once the client obtains a desired reference, it constructs a client-side representation of the object from the reference. This need not involve code within the application layer. For example, RPC services can provide clients with a means of constructing client-side handles from a composite name for the service, or from a reference containing an RPC address for the service. After getting this handle, the client performs all further operations on the object or service by supplying the handle.

This is the basic model of how the XFN interface is expected to be used. The XFN enterprise policies presented in Appendix D further encourage a bind/lookup model for how services and clients may rendezvous through the use of the naming service.

Applications may also use federated naming services indirectly through other interfaces. For example, consider a UNIX application that obtains a filename that it later supplies to the UNIX `open()` function. If the system provides XFN support for resolution of filenames, the application need not be aware that the strings it deals with are composite names rather than the traditional local pathnames. Some applications can thereby support the use of composite names without modification.

### 2.3.2 Basic Implementation Model

The XFN specifications do not dictate a specific way of providing support for the XFN interface. The implementor has great flexibility in terms of how to provide implementations of the XFN service for new and existing naming systems in different environments, based on different requirements in those environments. A potential implementor may find the basic model described below useful in order to understand how to provide the required functionality. This model and additional models are described in more detail in Appendix C, which provides the implementor guidelines on how different configurations of XFN with new and existing naming systems might be implemented.

The XFN client interface can be implemented in a layered manner. Figure 2-1 shows examples of this approach for a few existing naming services.

The top layer (labeled XFN API) is the XFN client interface with which callers interact directly. It implements the types of parameters used in the XFN interface.

In the bottom layer (labeled Context Implementation) are implementations of the XFN interface over specific naming services. Each such implementation over a specific naming service is based on knowledge of that naming service and essentially maps the XFN interface onto that service. It is expected that most of these implementations will work over existing naming services using the existing naming service interfaces and protocols.

Alternatively, the bottom layer could be a generic implementation that supports the XFN interface using a client/server model instead of a naming service-specific implementation. For example, there could be an XFN/DCE service to which the client invokes XFN requests. No

particular naming service-specific implementation or generic implementation is a mandatory component of XFN.

The middle layer (labeled XFN Framework) implements operations that span naming systems using interfaces presented in the bottom layer. This layer is responsible for guiding composite name resolution and invoking the proper operations for each context as required.

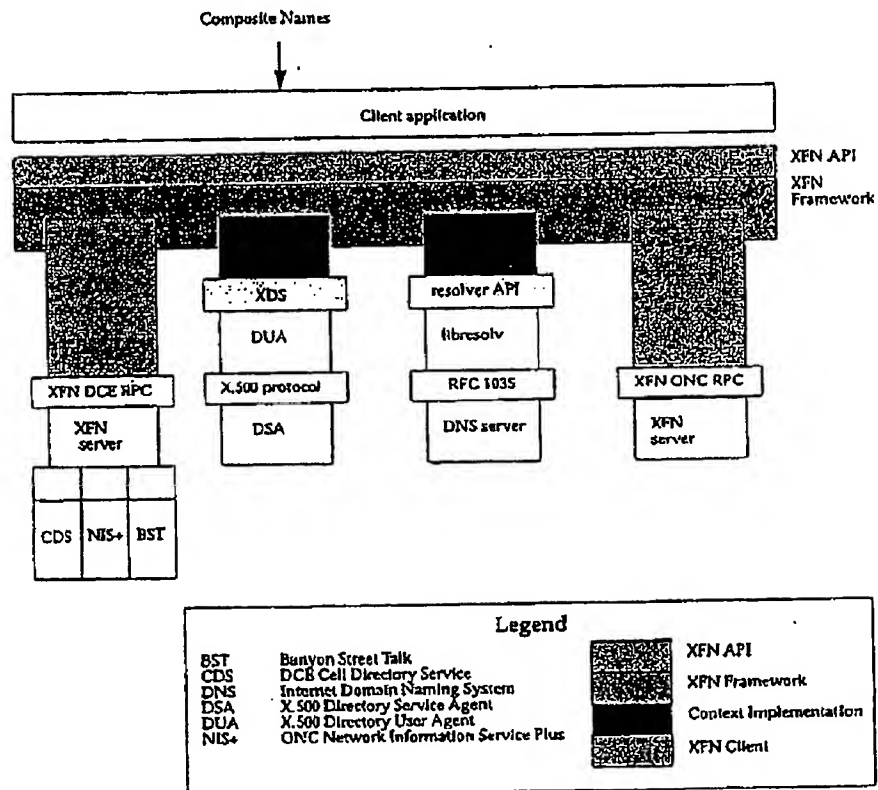


Figure 2-1 Example of an Implementation Model

## 2.4 Relationship of XFN to other Naming-related Services

### 2.4.1 Security

XFN does not define a security model nor a common security interface for contexts. Security-related operations, such as those required for authentication or access control, fall into the category of administrative interfaces which are expected to differ widely amongst federation members. Any single choice of security model at the XFN layer is prone to be fundamentally incompatible with the security provided by other direct interfaces, thereby giving rise to inconsistent protection that is susceptible to compromise. Consequently, the security administration interface is kept entirely separate from the federated naming service interface, outside the scope of XFN.

XFN does, however, provide means by which security can be integrated with specific XFN implementations. Operations that fail due to security-related problems can indicate in the status code the nature of the failure.

#### Authentication

Authentication is to be handled in the modules that implement the service interfaces for each particular member naming system. It occurs as part of the communication that occurs between the client and the naming service. Significant engineering issues remain when multiple security mechanisms and administrative domains are involved. These problems can be addressed on a one-to-one basis, or via a general federated security model, outside the scope of XFN.

XFN provides the status code `[FN_E_AUTHENTICATION_FAILURE]` to indicate that an operation failed due to authentication errors.

#### Access Control

Given the ability to authenticate the principal making a service request, a context service provider must then decide whether this principal should be granted or denied the request. Access control is to be handled through additional interfaces in the specific contexts. This can be done by having operations that control default authorisation at context creation and binding creation times, and having interfaces that modify the current authorisation settings. This is analogous to the *umask* and *chmod* scheme for POSIX.1 files.

The access control model is outside the scope of XFN.

XFN provides the status codes `[FN_E_CTX_NO_PERMISSION]` and `[FN_E_ATTR_NO_PERMISSION]` to indicate that an operation failed due to access control errors. In the case that the principal is not authorised to know that access has been denied due to permission problems, the status code `[FN_E_NAME_NOT_FOUND]` or `[FN_E_NO_SUCH_ATTRIBUTE]` is returned.

## *Guidelines for Federating a Naming System*

These guidelines are intended for naming system implementors who either want to integrate an existing naming system into the federation or develop a new XFN-conformant naming system (including any application that exports naming interfaces).

These guidelines are to be used in conjunction with the native specifications of the naming system to be federated and the implementation specification of the XFN client library.

The first section of this appendix describes some typical implementation models that an implementor can use to provide XFN service.

The remaining sections summarise information contained in Chapter 3 and Chapter 4. These contain guidelines to help integrate a naming system into the XFN federation. It is recommended that integrators publish specifications that define the behaviour of their XFN-conformant naming systems. This provides application writers that use XFN with necessary information for writing portable applications and administrators with the necessary information to integrate that naming system into the federation.

## C.1 Implementation Models

These guidelines do not prescribe any particular implementation model but in order to appreciate the features of the different possible configurations of the XFN system, it might be helpful to understand the various building blocks and its required functionality.

The three diagrams Figure C-1, Figure C-2 and Figure C-3 serve as examples of the conceptual models of the different possible configurations. The dark shaded boxes shown in these diagrams are building blocks that a naming service integrator needs to provide in order to integrate the naming system with XFN. The modules depicted in the three diagrams are defined as follows:

### XFN API

The *XFNAPI* is the complete set of interface operations defined in this XFN specification.

### XFN Framework

The *XFN framework* is the implementation of the XFN API, including the client library and the service provider interfaces necessary for integrating native naming systems.

### Context Implementation

The *context implementation* is the naming service-specific module on the XFN client system that is required to integrate legacy naming systems with XFN. The code of the context implementation is a wrapper that maps the XFN API to the API exported by the legacy naming system. The complexity of the context implementation depends on how well the XFN API maps to the native naming service API and which XFN operations are to be supported. At a minimum, the name resolution phase of all operations must be supported.

The techniques used to access the naming service-specific context implementations from the XFN framework may vary. For systems that support shared libraries and dynamic linking, a common approach might be that the context implementations are dynamically loadable modules.

This approach of integrating a naming service using a context implementation module does not require any modification to the existing naming service's source code nor does it require access to the naming service's source code. All that is needed is access to the module (library) that exports the naming service-specific API. This approach is by far the easiest and fastest way of adding an existing naming system into the XFN federation.

### XFN Client

The *XFN client* is a module that implements the client protocol machines for the XFN protocols.

Two XFN protocols are specified in Appendix A, the RPC based protocols for ONC+ systems (specified in RPCL) and for DCE environments (specified in IDL). The list of supported protocols might evolve over time.

In addition to supporting the protocols, the XFN client might provide services typically offered by naming clients, such as caching. The extent of this support is implementation-specific.

### XFN Protocol Exporter

The *XFN protocol exporter* is the module required on systems that export one of the XFN protocols. This could be a new naming system, an existing naming system that was modified to also support XFN protocols, or a system that supports the XFN client library and also exports XFN protocols (capable of acting as surrogate client).

The advantage for naming systems that support one of the specified protocols is that any existing XFN client that imports the protocols can be used to communicate with it. This is particularly useful for applications that need to export naming interfaces. Application

programmers do not have to duplicate the client-side implementation and they do not have to invent new naming interfaces. This provides additional benefits such as the ability to utilise caching and other mechanisms provided by the XFN client implementations, and a direct (and possibly more efficient) mapping of XFN operations to the application's naming operations.

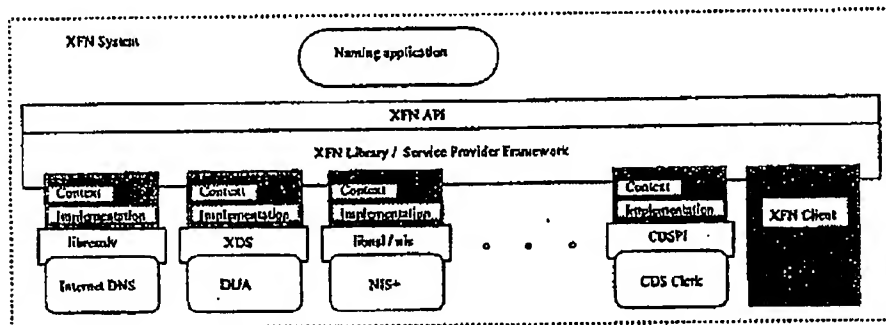


Figure C-1 XFN Configuration using Client Context Implementations

Figure C-1 shows the layering of the XFN client library on top of existing naming system clients on the same system. None of the legacy naming systems needs to be modified.

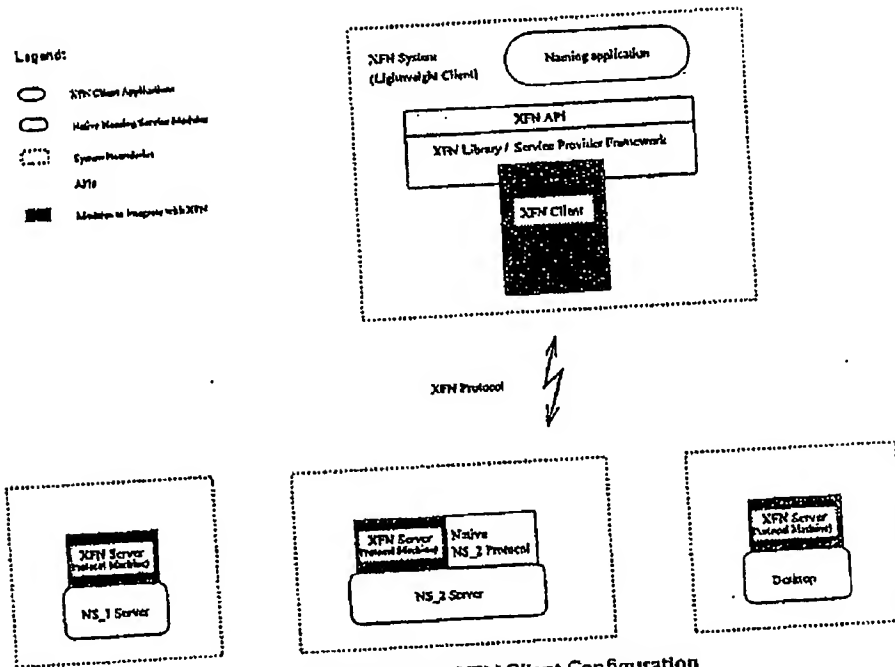


Figure C-2 Lightweight XFN Client Configuration

Figure C-2 shows multiple XFN systems that are connected via one of the specified XFN protocols. The client in this picture is a lightweight XFN client. The servers shown are name servers that directly export one of the specified XFN protocols.



The two modules shown in Figure C-3 are a lightweight XFN client and a server that acts as an intermediary. Similar to the client in Figure C-2, the client in Figure C-3 is a truly lightweight XFN client. None of the legacy naming system clients needs to be installed at that system. Depending on the client system's requirements, the XFN client can be implemented and configured to consume more or less resources, determined based on needs and availability. The XFN client might simply defer to mechanisms (such as for caching and replication) provided by the native naming system clients.

The legacy naming system clients in Figure C-3 reside on a remote system (similar to Figure C-1) that also exports at least one of the XFN protocols. This remote client can be viewed as a surrogate or proxy client that acts on behalf of the initial requestor and performs the native naming system functions.

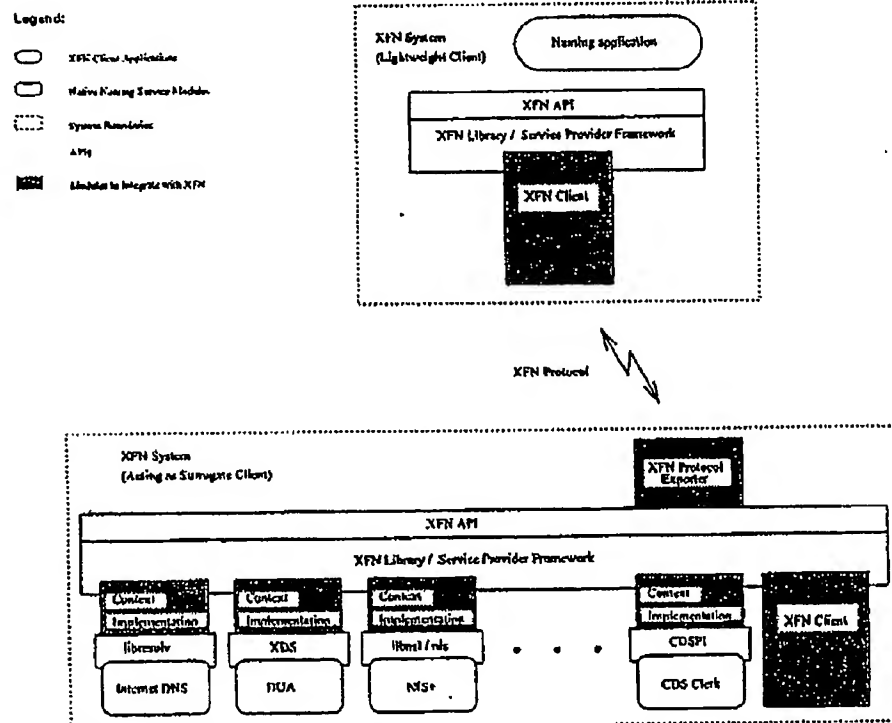


Figure C-3 XFN Configuration with Surrogate Client

Another aspect shown in Figure C-1 and Figure C-3 is the capability of the surrogate client to also import the XFN protocol (*XFN client module*). Such a configuration could serve emerging XFN servers or existing name servers that export one of the specified XFN protocols in addition to, or in replacement of, the native protocol.

Note that a *context implementation* precisely defines the set of modules that are co-located with the XFN framework to map the XFN API to the native naming service API. However, in the context of this appendix, we also use the term *context implementation* to mean the XFN mapping

## *Guidelines for Federating a Naming System*

### *Implementation Models*

code that is necessary at the server of a naming system that directly exports one of the XFN protocols (*XFN Protocol Exporter* and *XFN Server* in the diagrams).

## C.2 Federating with other Naming Systems

The role and potential configuration of the naming system to be federated must be determined. The naming system might be a global naming system, and therefore should provide some means for registering enterprises and possibly other application naming systems. Or, the naming system could be an enterprise-level or application-specific naming system.

Furthermore, it must be specified whether the naming system must always be joined at the root or whether other entry points are also supported.

Any naming system that participates in the federation must define whether other naming systems can be federated as subordinate naming systems. Naming systems in the application namespaces (such as file systems) might not permit further federation.

Subordinate naming systems are federated through *next naming system pointers* (*nns pointers*). There are two flavours of XFN *nns pointers*: *junctions* and *implicit*. Both may be supported concurrently in a single naming system.

The criteria for whether a naming system support junctions or implicit *nns pointers*, or both, are in Section 4.3 on page 63. For example, if a distinct name for naming the subordinate naming system is either not available (for example, already in use) or not desired (for example, it would not be a terminal name in that naming system), using implicit *nns pointers* is the preferred technique. An instance where junctions are better suited could be if subordinate naming systems must be selectable by explicit names. One might want to apply the rule of the thumb that implicit *nns pointers* are preferred for the global namespace (because X.500 and DNS names cannot be guaranteed to be leaf names) and most enterprise naming systems are expected to support junctions.

### C.2.1 Junctions

Junctions are terminal names in a naming system that hold the reference information of subordinate naming system contexts.

The context implementation defines how references are constructed from information that is associated with the named entry. A common approach might be to use attributes for storing the appropriate reference and address information. Implementations might or might not permit the use of *bind* and *unbind* operations to create and delete references of junctions.

If junctions are supported, the context implementation must ensure the correct behaviour of XFN operations on junctions. In particular, the name resolution phase in XFN operations must follow a junction and resolve to the context that is bound by the reference of the junction.

### C.2.2 Implicit Next Naming System Pointers

If the context implementation supports implicit next naming system pointers for federating naming systems, the context implementation must determine how references are managed.

The context implementation must ensure the correct behaviour of XFN operations on implicit *nns pointers*. If a name resolution is performed on a name that has a trailing XFN component separator, the implicit *nns pointer* must be followed.

### 3.3 Name Syntax

For naming systems participating in the naming federation it is necessary to specify the syntax of the compound name and to specify if the context supports the weak or strong separation model.

#### C.3.1 Weak and Strong Separation

If the component separator for atomic names of the naming system is distinct from the XFN component separator ('/') and if this XFN component separator character is not used in the compound name in unescaped and unquoted form, only the strong separation model applies.

If the atomic name separator is the same as the XFN component separator and the ordering of atomic names is not left-to-right, strong separation must be enforced by either quoting the compound name or escaping the atomic name separators. It must be specified which rule — quoting or escaping — applies (possibly both).

If the atomic name separator is the same as the XFN component separator and the ordering of atomic names is left-to-right, either strong or weak separation might be supported, or both. Please note, that in the instance where both forms of separation are supported, the context implementation must be prepared to receive in a XFN component name either a full compound name or a single atomic name. Weak separation can only be supported if at least one of the conditions that are specified in Section 4.2 on page 60 apply, namely the naming system must:

- be a terminal naming system
- if not terminal, the context must be able to do a syntax-specific discovery of naming system boundaries
- if not terminal, the context must be able to return the remaining unresolved components.

It must be specified which rule applies. If syntax-specific checking is done, the restriction for the subordinate naming systems must be clearly specified in order to avoid the use of (top level) names that conflict with the syntax rules. For instance, if the name syntax defines typed names for the atomic name component that uses the equal character ('=') as the type separator, the first name component of the subordinate naming system must not contain an unescaped equal character.

#### C.3.2 Syntax Attributes

In order to permit applications to use the operations on compound names, the context implementation must provide for a means of supporting the interface function for retrieving the syntax attributes (`fn_ctx_get_syntax_attrs()`). If the "fn\_syntax\_type" attribute indicates the support of the XFN standard model, it can be assumed that the XFN framework implementation provides the support for interpreting the syntax attribute values. If the XFN standard model is not supported, the context implementation must provide the appropriate module for evaluating the syntax attributes and for compound name resolution. (Refer to Section 3.8 on page 50, for more information on the syntax attributes and the XFN standard model).

If the syntax attributes are fixed for a given naming system, the context implementation might not provide interfaces for modifying these attributes; in fact they might not be implemented as regular attributes of the naming system. It should be specified how these attributes are determined and whether they can be set and modified (through XFN attribute operations, for instance).

## C.4 Context Operations

A conformance statement for the federated naming system must specify the level of support of the XFN context operations. The minimum requirement for XFN conformance is the support of the *fn\_ctx\_lookup()* operation and the support of the name resolution phase of the other operations.

Depending on the semantics of operations in the underlying naming system and the desired level of integration, other context operations might also be supported.

Operations that are not supported still must be provided with implementations that perform at least the name resolution phase to the target context. If the target context does not support the actual operation, the implementation returns the status [FN\_E\_OPERATION\_NOT\_SUPPORTED].

The context implementation may support operations such as *fn\_ctx\_list\_names()*, *fn\_ctx\_list\_bindings()* and *fn\_ctx\_create\_subcontext()*, but the native naming system might not provide these as atomic operations. The atomicity guarantees for such operations must be specified.

There are a number of context operations whose behaviour depends on the semantics of operations in the underlying naming system. The implementation specifications must specify how the XFN operations effect the state of the underlying naming system. The XFN context operations that might incur different behaviour on different naming systems are detailed in the following subsections (For details on the operation's semantics, refer to the manual reference pages in Chapter 6.):

### *fn\_ctx\_bind()*

Naming systems might or might not support the exclusive flag.

Naming systems might or might not permit binding a name without also creating some attributes first. Also, the *fn\_ctx\_bind()* operation might only permit creation of certain types of objects in the namespace.

If next naming system pointers are supported, implementations might not permit binding of these references through the *fn\_ctx\_bind()* operation.

### *fn\_ctx\_unbind()*

The failure semantics of this operation is different depending on the type of object in the namespace. For example, an *fn\_ctx\_unbind()* on some contexts might not be permitted if it is not terminal (for example, if it is a directory).

### *fn\_ctx\_create\_subcontext()*

Similar to *fn\_ctx\_bind()*, some naming systems might not permit creating a subcontext without also creating some attributes first.

If naming systems support namespace partitioning or replicated servers, the *fn\_ctx\_create\_subcontext()* might not provide the sufficient information to perform the creation.

### *fn\_ctx\_destroy\_subcontext()*

Similar to *fn\_ctx\_unbind()*.

### *fn\_ctx\_rename()*

The scope of the *fn\_ctx\_rename()* operation must be specified. Some naming systems might have restrictions on the semantics of rename. For example, only renames within the same context or on the same server might be permitted.

*Context Operations*

*fn\_ctx\_lookup\_link()*

The implementation of XFN links must be specified (the information could possibly be stored in specific attributes). Also, the relationship of XFN links to any support for native soft links or aliases in the underlying naming system needs to be specified.

*fn\_ctx\_handle\_from\_ref()*

The implementation of the XFN framework in conjunction with the different context implementations determine the details of this operation. It is the responsibility of specific context implementations to define the properties of the `FN_ctx_t` object that is returned and to maintain the context handle and appropriate state information.

*fn\_ctx\_get\_ref()*

Implementations may vary in their support for this operation in that the returned reference may contain a list of addresses that is different from the originally supplied to `fn_ctx_handle_from_ref()`.

## C.5 Attribute Operations

Similar to the context operations, a conformance statement for the federated naming system must specify the level of support of the XFN attribute operations.

The operations on attributes might be partially, fully, or not supported. Typically, partial support would contain the operations on single attributes and exclude the multi-attribute operations.

Operations that are not supported still must be provided with implementations that perform at least the name resolution phase to the target context. If the target context does not support the operation, the function returns the status `[FN_E_OPERATION_NOT_SUPPORTED]`.

The context implementation may support operations such as `fn_attr_get_ids()`, `fn_attr_multi_get()` and `fn_attr_multi_modify()` but the native naming system might not provide these as atomic operations. The atomicity guarantees for such operations must be specified.

If the attribute operations are at least partially supported, the implementation specification of the naming service must specify how its attribute model maps to the XFN attribute operations. Section 3.3.2 on page 27 discusses that the attribute model is not dictated by XFN. Some naming systems might associate attributes with the named objects, others might support attributes that are associated with names in contexts that are bound to objects, or naming systems support a combination of both. Some naming systems might not even distinguish between the two models.

Implementations might only support the attribute operations for certain types of attributes or attributes that are associated with certain type of entries in the namespace (for instance, directories or soft links may be excluded).

For the attribute modify operations, implementations must determine how the operation codes `FN_ATTR_OP_ADD`, `FN_ATTR_OP_ADD_EXCLUSIVE`, `FN_ATTR_OP_REMOVE`, `FN_ATTR_OP_ADD_VALUES` and `FN_ATTR_OP_REMOVE_VALUES` apply and what the exact semantics are.

### C.5.1 Attributes and Next Naming System Pointers

In order to support next naming system pointers, the implementation specification must define how access to objects that the next naming system pointers is bound to (the contexts of the subordinate naming systems) can be disambiguated from access to attributes in the superior naming system. This disambiguation is particularly necessary for junctions where attributes are maintained in both the name of the junction in the superior naming system and the root context of the subordinate naming system.

If implicit *nns* pointers are used and one wants to access attributes that are associated with the subordinate naming system context, a trailing slash (XFN component separator) used for the resolution of the context disambiguates the access. If a trailing slash is passed in the *name* argument of an attribute operation, an implementation might support operations on attributes that are associated with the implicit *nns* pointer. (The analogy to the behaviour of junctions becomes obvious if one considers a trailing slash as a trailing empty name — this *nns* pointer is implicit).

For junctions, no trailing XFN component separators are permitted, but one typically accesses attributes associated with the subordinate naming system context if the context of the fully resolved name is passed as an argument to the attribute operation (the *name* argument is empty). In contrast, if the name of the junction is passed in the *name* argument, one might access attributes associated with the junction name (this might or might not be supported by naming system implementations).

Attribute Operations

Figure C-4 and Figure C-5 demonstrate possible associations between attributes, objects and names in a naming system. These diagrams show how one might use attributes to maintain references for next naming system pointers.

In the first example, Figure C-4, the reference of an implicit next naming system pointer is supported in the global namespace as an attribute that is associated with a context object.

If we take the name `.../umass.edu/` as an example (both with and without trailing slash), we find that there are four possible ways how this name can be represented to the attribute operations, depending on what the starting context (the object passed to the `ctx` argument) is set to:

ctx of	name	# in Diagram	Attribute Operation Refers To
<code>.../edu</code>	<code>umass</code>	(1)	Attributes associated with name <code>umass</code> in the context of <code>edu</code>
<code>.../umass.edu</code>	<code>empty</code>	(2)	Attribute associated with the context of <code>umass</code>
<code>.../umass.edu</code>	<code>/*</code>	(3)	Attributes associated with anonymous <code>ns</code> pointer maintained in context <code>umass</code>
<code>.../umass.edu/</code>	<code>empty</code>	(4)	Attributes associated with root context of next naming system

The second example, Figure C-5, depicts a possible implementation for junctions. The junction's reference is maintained in an attribute (or conceivably set of attributes) that is associated with a terminal name of the enterprise naming system.

If we take the name `.../umass.edu/science/service/spreadsheet` as an example for naming a junction, we have two ways of passing this to one of the attribute operations (note that trailing slashes are not permitted for junctions):

ctx of	name	# in Diagram	Attribute Operations Refers To
<code>service</code>	<code>spreadsheet</code>	(5)	Attributes associated with junction name <code>spreadsheet</code> in the context of <code>service</code>
<code>service/spreadsheet</code>	<code>empty</code>	(6)	Attributes associated with root context of next naming system



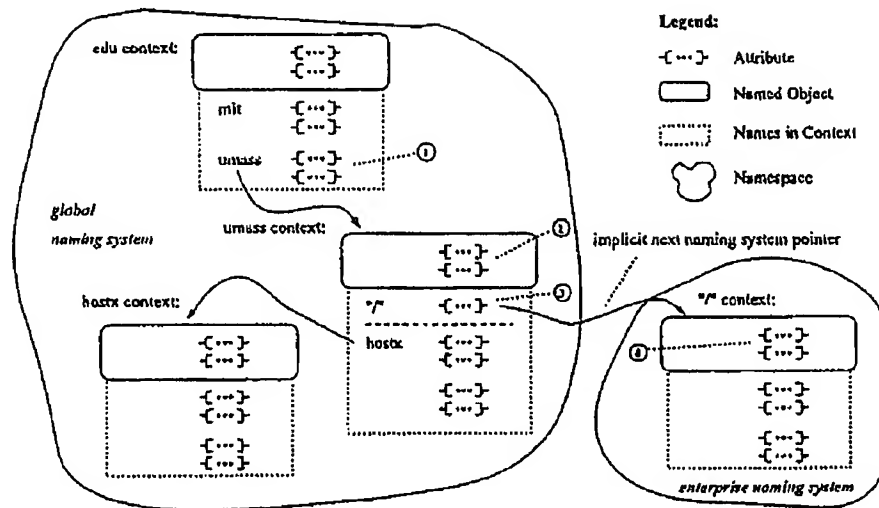


Figure C-4 Attribute Example with Implicit Next Naming System Pointer

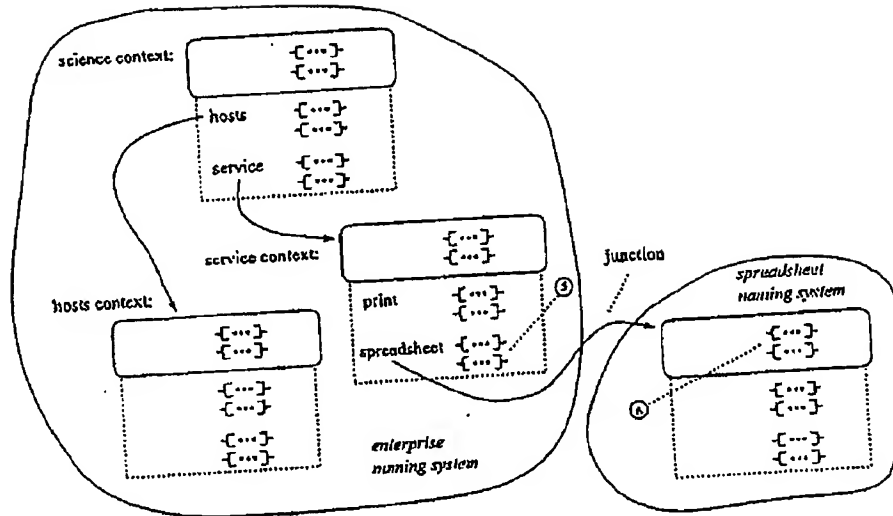


Figure C-5 Attribute Example with Junction

## **C.6    Reference and Address Types and its Registration**

The values and encodings of reference and address types exported by the naming system need to be specified in order to support its federation by other naming systems. The implementor of the context implementation must decide which of these types are necessary to be registered with X/Open.

Generally, a single distinct reference type is specified for uniquely identifying a particular naming service. This reference type is used by the XFN framework as the discriminator for locating and dispatching to the context implementation.

XFN framework implementations may also permit default fallbacks where reference type specific implementations are not available. The address types can be used as discriminator instead. If a reference contains multiple address types, the order of selection is unspecified.

In cases where servers of a naming system directly export one of the XFN protocols, the appropriate reference types (for example, `FN_DCE_SERVICE_REF`) are used for selecting the client implementation. If implementations for the reference types are available, the XFN framework dispatches to the XFN client module that imports these protocols.

It might also be relevant for users and administrators of a federated naming system to know how XFN references and addresses map to its representation in the naming system. Since these might typically be stored in attributes, it is necessary for implementors of a federated naming system to define the format and encodings.

## *Guidelines for Federating a Naming System*